

UNIVERSITAT
JAUME·I

GRADO EN INGENIERÍA INFORMÁTICA

TRABAJO FINAL DE GRADO

**Desarrollo de un sistema de entrega de
diagnósticos radiológicos a pacientes y médicos
referentes aplicado a centros de radiología**

Realizado por:

Denis HERNÁNDEZ ESTEBAN

Supervisado por:

Rafa FORCADA MARTÍNEZ

Tutorizado por:

M^a Ángeles LÓPEZ MALO

Fecha de lectura: 21 de Julio de 2015

Curso académico 2014/2015

Resumen

El presente documento se corresponde con la memoria del Trabajo Final de Grado realizado en la empresa ActualTec Innovación Tecnológica, S.L. Mi trabajo ha consistido en el desarrollo de un sistema de entrega y visualización de diagnósticos radiológicos a pacientes y médicos referentes integrado en la plataforma *online* Actualpacs.

La radiología es la especialidad médica que se ocupa de generar imágenes del interior del cuerpo por medio de rayos X, ultrasonido y campos magnéticos, y de utilizarlas para el diagnóstico y, en menor medida, para el pronóstico y el tratamiento de las enfermedades.

El sistema pretende hacer llegar estas imágenes médicas de forma *online* a los pacientes y médicos, para que puedan consultarlas e informar de manera remota sin estar conectados a internet.

Palabras clave

Teleradiología, PACS, DICOM, ejecutable al vuelo, encapsulado.

Keywords

Teleradiology, PACS, DICOM, executable on the fly, encapsulated.

Índice general

1. Introducción	9
1.1 Descripción de la empresa	9
1.1.1 Datos de la empresa	9
1.1.2 Actividad de la empresa	9
1.2 Contexto y motivación del proyecto	10
1.3 Objetivos del proyecto	10
2. Descripción del proyecto	13
2.1 Descripción del proyecto	13
2.2 Conceptos clave del proyecto	14
2.2.1 Teleradiología	14
2.2.1 PACS	14
2.2.2 DICOM	15
2.2.3 Ejecutable al vuelo	16
2.2.4 Encapsulado	16
2.3 Alcance	16
2.4 Restricciones	17
3. Planificación del proyecto	21
3.1 Metodología y definición de tareas	21
3.1.1 Tareas principales	21
3.2 Planificación temporal de las tareas	23
3.2.1 Diagrama de Gantt	24
3.2.2 Horario de trabajo	24
3.3 Estimación de recursos del proyecto	25
3.3.1 Software	25
3.3.2 Hardware	25
3.3.3 Humanos	25
3.3.4 Otros	26
3.3.5 Resumen	26
4. Análisis y diseño del sistema	29
4.1 Análisis de requisitos	30
4.1.1 Ejecutable EXE	30
4.1.2 Ejecutable DMG	31
4.1.3 Integración del sistema en la plataforma Actualpacs	32
4.2 Diseño del sistema	33
4.2.1 Preparación del ejecutable EXE	34
4.2.2 Preparación del ejecutable DMG	37
4.2.3 Integración del sistema en la plataforma Actualpacs	39
4.2.4 Diseño de la interfaz de la plataforma Actualpacs	40

5. Implementación, pruebas y documentación	43
5.1 Tecnología empleada.....	43
5.1.1 Desarrollo del proyecto	43
5.1.2 Creación y personalización del EXE	44
5.1.3 Creación y personalización del DMG.....	44
5.1.4 Integración en Actualpacs	45
5.2 Detalles de implementación	46
5.2.1 Ejecutable EXE.....	46
5.2.1.1 Pruebas unitarias ejecutable EXE.....	47
5.2.1.2 Certificado de firma de código digital.....	48
5.2.2 Ejecutable DMG.....	50
5.2.2.1 Pruebas unitarias ejecutable DMG	51
5.2.3 Integración del sistema en la plataforma Actualpacs.....	51
5.2.3.1 Instalar Laravel y preparar entorno de desarrollo.....	52
5.2.3.2 Añadir nueva columna y botón Descargar	54
5.2.3.3 Asociar al botón la acción de descargar ejecutable.....	55
5.2.3.4 Analizar tiempo de generación y descarga del ejecutable.....	57
5.2.3.5 Mostrar notificación al descargar	58
5.2.3.6 Adaptar sistema de generación y descarga del ejecutable.....	59
5.2.3.7 Controlar errores.....	61
5.2.3.8 Comprobar ajustes de configuración	62
5.3 Validación y pruebas	64
5.4 Puesta en marcha	68
5.5 Documentación.....	69
6. Conclusiones.....	73
7. Bibliografía.....	75
Anexos	79
Anexo I. Planificación.....	79
Anexo II. Código fuente.....	80

Capítulo 1

Introducción

1.1 Descripción de la empresa

1.2 Contexto y motivación del proyecto

1.3 Objetivos del proyecto

1. Introducción

En este primer capítulo se muestra la descripción de la empresa, el contexto y la motivación para llevar a cabo el proyecto, así como los objetivos principales.

1.1 Descripción de la empresa

1.1.1 Datos de la empresa

- Nombre de la empresa: ActualTec Innovación Tecnológica, S.L.
- Dirección: Avda. Vicent Sos Baynat s/n, Universidad Jaume I Parque Científico, Tecnológico y Empresarial (Espaitec 2)
- C.P: 12006
- Localidad: Castellón de la Plana, España
- Teléfono de contacto: +34 964 199 495
- Personas de contacto: Sergio Fabra Llopis y Rafa Forcada Martínez
- Tutor de la empresa: Rafa Forcada Martínez

1.1.2 Actividad de la empresa

ActualTec es una empresa de base tecnológica fundada en 2007 por dos ingenieros en informática titulados por la Universitat Jaume I y cuenta en la actualidad con dos líneas de negocio fuertemente consolidadas.

Por una parte, ActualWeb, que se dedica al desarrollo y mantenimiento web, así como al alojamiento web y gestión de dominios. Provee de un servicio de hosting propio con el cual abastece tanto a todos sus clientes como así mismo. Y por otra parte ActualMed, que es el área donde he estado realizando mi estancia en prácticas y proyecto fin de grado. Ésta se dedica a ofrecer soluciones en radiología y teleradiología. Además, diseña y desarrolla soluciones de gestión de imagen médica digital para mejorar la calidad del diagnóstico médico.

El equipo está formado por 10 trabajadores que desarrollan proyectos de carácter nacional e internacional.

Entre sus principales clientes destacan hospitales, clínicas radiológicas y médicos radiólogos por una parte, y empresas de desarrollo y alojamiento web, así como pymes con presencia *online* y servicios de email, por la otra.

1.2 Contexto y motivación del proyecto

Uno de los productos que ofrece la línea de negocio ActualMed es un sistema RIS/PACS, llamado Actualpacs [1], que permite almacenar en la nube las imágenes e informes médicos que han elaborado los especialistas, mejorando la productividad y la calidad del diagnóstico radiológico. Gracias a su acceso *online*, se puede visualizar, informar y compartir casos de interés para segundas opiniones, desde cualquier lugar, eliminando las restricciones espaciales y temporales.

En la actualidad, Actualpacs proporciona a los especialistas la posibilidad de visualizar las imágenes médicas y la generación de informes tras el análisis de las imágenes, además de otras muchas más funcionalidades adicionales que les ayudan a realizar su trabajo de una manera más eficiente. Sin embargo, no ofrece la posibilidad de descargar las imágenes de los estudios y que éstas puedan ser visualizadas en local, sin estar conectado a internet ni tener instalado ningún software.

La motivación del proyecto viene dada porque algunas clínicas radiológicas con las que trabaja la empresa, tienen pacientes que solicitan el poder descargar sus estudios para poderlos visualizar en sus ordenadores personales o guardarlos en un dispositivo de almacenamiento, para poderlos transportar y consultar permanentemente sin la dependencia de estar conectados a internet.

Por otro lado, este sistema también sería de utilidad para los médicos radiólogos en el caso de que los visores de imágenes médicas disponibles en Actualpacs dejaran de funcionar y tuvieran que realizar un diagnóstico.

1.3 Objetivos del proyecto

El principal objetivo de este proyecto es desarrollar una de las partes que constituyen el sistema de teleradiología para entregar los resultados radiológicos de forma *online* y permitir que, de forma *offline*, los pacientes puedan consultar sus estudios y los radiólogos dispongan de otro modo de realizar el diagnóstico.

El objetivo principal se puede desglosar en los siguientes subobjetivos:

- Realizar un sistema que genere un ejecutable “al vuelo”.
- Integrar este sistema en la plataforma *online* Actualpacs.
- Favorecer el acceso a los estudios a los pacientes.
- Facilitar el diagnóstico radiológico.

Capítulo 2

Descripción del proyecto

1.1 Descripción del proyecto

1.2 Conceptos clave del proyecto

1.3 Alcance

1.4 Restricciones

2. Descripción del proyecto

En el siguiente capítulo se describe el proyecto, los conceptos clave sobre los que se sustenta, su alcance y las restricciones que existen para su realización.

2.1 Descripción del proyecto

El propósito de este proyecto consiste en desarrollar un sistema que permita entregar las imágenes médicas DICOM [2] de los estudios a los pacientes y médicos referentes de forma *online*.

En la plataforma *online* Actualpacs, habrá que añadir un botón por cada estudio, que permita generar un ejecutable “al vuelo” y que al descargarlo y ejecutarlo en el ordenador del paciente o del radiólogo, se abran las imágenes del estudio seleccionado en un visor de imágenes médicas, sin necesidad de que éstos lo tengan instalado previamente en su ordenador personal.

Un sistema como éste permitiría que el tiempo dedicado a realizar un diagnóstico radiológico disminuyera y que tanto el paciente como el radiólogo, una vez descargado el estudio, pueda visualizarlo desde cualquier lugar sin necesidad de tener una conexión a internet. Con esto también evitaremos segundas visitas innecesarias para el paciente y favorecemos que éste pueda almacenar permanentemente su estudio o grabarlo en una memoria USB o CD y visualizarlo en cualquier momento.

Se generará un ejecutable distinto dependiendo del sistema operativo desde el cual acceda el usuario a Actualpacs. En el caso de que sea Windows, se generará un archivo “.exe” y en el de Mac una imagen “.dmg”.

Para que el usuario pueda visualizar las imágenes del estudio seleccionado, el ejecutable deberá incluir un visor DICOM de imágenes médicas, además de éstas. Asimismo, cuando se ejecute el ejecutable, el visor deberá abrirlas automáticamente.

El visor también dependerá del sistema operativo. La empresa ha decidido que para Windows sea el RadiAnt DICOM Viewer y para Mac, el OsiriX DICOM Viewer.

2.2 Conceptos clave del proyecto

2.2.1 Teleradiología

La teleradiología es un novel método que consiste en la transmisión electrónica de imágenes radiológicas. Consiste en la aplicación de tecnologías y recursos que posibilitan la interpretación de imágenes médicas obtenidas en lugares remotos al lugar de interpretación.

Constituye uno de los recursos más importantes para el paciente, ya que por esta vía reciben un diagnóstico oportuno elaborado por médicos referentes, que son los médicos especializados en este campo.

Debido a la escasez y concentración de médicos radiólogos en nuestro país, la teleradiología supone el recurso más viable para que los pacientes reciban un diagnóstico oportuno por médicos especialistas y subespecialistas.

Los hospitales y clínicas adquieren, automáticamente, cobertura de todos sus horarios, días hábiles y festivos, independiente de la disponibilidad de médicos radiólogos en la zona, volumen de estudios, vacaciones y días libres, reduciendo costes y facilitando la gestión de un departamento de radiología de cualquier magnitud.

2.2.1 PACS

Un servidor PACS [3] es un sistema de almacenamiento digital, transmisión y descarga de imágenes radiológicas. Los sistemas PACS se componen de partes software y hardware, que directamente se comunican con el equipamiento de las distintas modalidades (medicina nuclear, ultrasonido, resonancia magnética, etc.) y obtienen las imágenes de éstas. Las imágenes son transferidas a una estación de trabajo (Workstation) para su visualización y emisión de informes radiológicos. El visor PACS es un software que se instala en la estación de trabajo y que utiliza el radiólogo para recibir y mostrar las imágenes radiológicas. Las imágenes son archivadas entonces en el servidor PACS para su descarga posterior hacia las estaciones de trabajo.

La digitalización de la imagen radiológica ha permitido la distribución en un formato digital de las tradicionales placas. Este hecho junto con el uso de los servidores PACS ha proporcionado a los radiólogos y técnicos de la imagen la posibilidad de acceder a las mismas desde cualquier lugar en cualquier momento.

Los componentes básicos de un sistema PACS son:

- Servidor Central PACS: Se compone del hardware principal del sistema.
- Estación de trabajo PACS: Permite a los radiólogos la visualización y análisis de las imágenes digitales.
- Sistema de Base de Datos: Se encarga de gestionar el almacén de toda la información e imágenes del sistema PACS.
- Servidor DICOM: Responsable de toda la comunicación DICOM con las modalidades de imagen (como por ejemplo Tac o Resonancia Magnética), otros servidores PACS y estaciones de trabajo DICOM.
- Sistema de almacenamiento: Es el soporte físico requerido para almacenar las imágenes DICOM del sistema PACS.
- Interfaces a RIS/HIS: Consolida toda la información del paciente desde diferentes fuentes, lo que permite un flujo de trabajo idóneo.
- Servidor Web para Acceso Remoto: Imprescindible para teleradiología. Mediante el acceso Web, las imágenes e información almacenadas en el servidor PACS pueden ser accedidas mediante un navegador web, como por ejemplo Internet Explorer, Mozilla Firefox, Safari, etc.

2.2.2 DICOM

DICOM (Digital Imaging and Communications in Medicine) es un formato universal para el intercambio de imagen médica digital, pensado para su manejo, visualización, almacenamiento, impresión y transmisión. Todos los sistemas PACS, modalidades y estaciones de trabajo que se comuniquen entre sí, deben hacerlo bajo este estándar.

El estándar DICOM incluye la definición de un formato de fichero y de un protocolo de comunicación de red. En cuanto al formato, aparte de la propia imagen también contiene una serie de metadatos asociados a ésta como puede ser la información del paciente (nombre, sexo, identificación, etc.), la modalidad, el proceso de adquisición de la imagen, etc. Como estándar de comunicación, DICOM define un conjunto muy amplio de servicios, la mayoría de los cuales implica la transmisión de datos sobre la red. El protocolo de comunicación es un protocolo de aplicación que usa TCP/IP para la comunicación entre sistemas.

2.2.3 Ejecutable al vuelo

Un ejecutable es un archivo que tiene la capacidad de poder ser ejecutado de forma independiente, o en otras palabras, que no necesita ser ejecutado por una aplicación externa. Estos archivos son ejecutados y controlados por el sistema operativo. Contiene un programa, y generalmente necesitan de otros archivos para funcionar (aunque no es necesario).

En concreto, un ejecutable al vuelo es un archivo con las características mencionadas anteriormente que se genera automáticamente al realizar una acción. En este caso, se generará un EXE o un DMG al pulsar un botón.

2.2.4 Encapsulado

En informática, el encapsulamiento consiste en ocultar la implementación y los atributos de un objeto, de manera que sólo se puede cambiar su estado mediante ciertas operaciones definidas en la interfaz del objeto.

Por analogía, llamaremos Encapsulado a la carpeta que contendrá los archivos necesarios para la generación y funcionamiento del ejecutable, el visor y las imágenes del estudio, a partir de la cual se creará, debido a que el usuario del ejecutable puede obviar cómo está implementado y cuál es su contenido para simplemente centrarse en utilizarlo.

2.3 Alcance

El alcance del sistema define los límites desde el punto de vista organizativo y funcional.

- Organizativo: el sistema ha de tener interacción con los centros sanitarios, áreas de radiología, radiólogos y pacientes.
- Funcional: el sistema tiene que cubrir la funcionalidad del área de radiología de un centro, concretamente del proceso de entrega y visualización de los diagnósticos. Por tanto, el sistema tiene que:
 - Dar soporte para la entrega de los estudios
 - Dar soporte para la visualización de las imágenes DICOM
 - Facilitar el diagnóstico en caso de que fallen los visores *online* de Actualpacs

2.4 Restricciones

Las restricciones informáticas para desarrollar este proyecto residen en la necesidad de disponer de un PC con Microsoft Windows, una máquina virtual con la distribución de Linux CentOS y un Mac con OS X.

En cuanto a las restricciones temporales, cabe destacar que existe una limitación de tiempo de 300 horas para la realización del proyecto.

Por otra parte no existen restricciones con respecto al software o hardware, si bien se ha de tener en cuenta que es necesario disponer de una conexión a internet para acceder a la plataforma *online* Actualpacs.

La velocidad con la que se generará el ejecutable dependerá de la cantidad de imágenes médicas que contenga el estudio seleccionado y la descarga dependerá de la velocidad de la conexión, así pues, siempre será mejor disponer de una conexión de alta velocidad.

Capítulo 3

Planificación del proyecto

3.1 Metodología y definición de tareas

3.2 Planificación temporal de las tareas

3.3 Estimación de recursos del proyecto

3. Planificación del proyecto

En este apartado se muestra la planificación seguida durante la realización del proyecto. La lista de tareas, la planificación temporal, y las dependencias entre ellas, así como una estimación inicial de los recursos software, hardware y humanos que se requerirán para el desarrollo del proyecto.

3.1 Metodología y definición de tareas

Para tratar de alcanzar los objetivos que se persiguen con el desarrollo de este proyecto será necesario llevar a cabo tres tareas principales que a su vez se desglosarán en las siguientes dos fases principales:

La primera de ellas consiste en la definición de requisitos, análisis y diseño. En esta fase nos centramos en documentar y establecer las bases del proyecto.

En la segunda fase nos encontramos con el desarrollo y puesta en marcha, en la que se realizará la implementación, validación y testeo del sistema.

3.1.1 Tareas principales

- **T1 - Generar un ejecutable para Windows:** desarrollar un sistema que permita crear automáticamente un archivo EXE que cumpla con los requisitos mencionados en el capítulo anterior y que pueda ser personalizado al gusto.
- **T2 - Generar un ejecutable para Mac:** desarrollar un sistema que permita crear automáticamente una imagen DMG que cumpla con los requisitos mencionados anteriormente y que pueda ser personalizada.
- **T3 - Integrar el sistema que genera un ejecutable en Actualpacs** para que los pacientes y radiólogos se beneficien de esta nueva utilidad.

En primera instancia se estimó que únicamente daría tiempo a desarrollar el sistema que permitiese generar un ejecutable “al vuelo” y que posiblemente integrarlo dentro de ActualPacs pasaría a ser responsabilidad de los desarrolladores de la empresa. Por lo que inicialmente se hizo una planificación sin tener en cuenta la parte de integración. En la Tabla 1 se puede observar el desglose de las tareas con esta planificación inicial del proyecto.

Id	Nombre de tarea	Duración
1	Trabajo Final de Grado	300 horas
2	Desarrollar la propuesta técnica	50 horas
3	Inicio	12 horas
4	Definir el proyecto con el tutor y el supervisor	2 horas
5	Definir método de trabajo y documentación	5 horas
6	Definir formato y estándares de trabajo	5 horas
7	Documentar y planificar el proyecto	15 horas
8	Revisar el contexto y buscar información	10 horas
9	Identificar alcance y objetivos	5 horas
10	Planificar el proyecto	23 horas
11	Definir tareas y estimar fechas	4 horas
12	Crear diagrama de Gantt	4 horas
13	Documentar la propuesta del proyecto	15 horas
14	Entregar Propuesta técnica	0 horas
15	Desarrollo técnico del proyecto	250 horas
16	Instalar máquina virtual	5 horas
17	Instalación Virtual Box	1 hora
18	Instalación distribución Linux	4 horas
19	Crear ejecutable para Windows	100 horas
20	Análisis	15 horas
21	Diseño	30 horas
22	Personalización	10 horas
23	Desarrollo	40 horas
24	Pruebas	5 horas
25	Crear ejecutable para Mac	135 horas
26	Análisis	25 horas
27	Diseño	40 horas
28	Personalización	15 horas
29	Desarrollo	50 horas
30	Pruebas	5 horas
31	Elaboración de un manual para los programadores	10 horas
32	Documentación y presentación del TFG	135 horas
33	Redacción de informes quincenales	10 horas
34	Redacción de la memoria técnica	90 horas
35	Entrega de la memoria técnica	0 horas
36	Preparación de la presentación oral	34 horas
37	Presentación oral	1 hora

Tabla 1. Desglose inicial en tareas

3.2 Planificación temporal de las tareas

Al cabo de unas de unas semanas y tras haber avanzado bastante el proyecto, nos dimos cuenta de que posiblemente si que daría tiempo a integrarlo, quedando la planificación temporal de las tareas como se muestra en la Figura 1.

A través de ésta resulta fácil identificar las dependencias entre tareas y determinar la fecha de finalización de las mismas.

Id	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras	Nombres de los recursos
1	Trabajo Final de Grado	300 horas	lun 13/10/14	vie 09/01/15		
2	Desarrollar la propuesta técnica	50 horas	lun 13/10/14	vie 24/10/14		
3	Inicio	12 horas	lun 13/10/14	mié 15/10/14		
4	Definir el proyecto con el tutor y el supervisor	2 horas	lun 13/10/14	lun 13/10/14		Analista;Diseñador;Programador
5	Definir método de trabajo y documentación	5 horas	lun 13/10/14	mar 14/10/14	4	Analista;Diseñador;Programador
6	Definir formato y estándares de trabajo	5 horas	mar 14/10/14	mié 15/10/14	5	Analista;Diseñador;Programador
7	Documentar y planificar el proyecto	15 horas	mié 15/10/14	lun 20/10/14		
8	Revisar el contexto y buscar información	10 horas	mié 15/10/14	vie 17/10/14	3	Analista
9	Identificar alcance y objetivos	5 horas	vie 17/10/14	lun 20/10/14	8	Analista
10	Planificar el proyecto	23 horas	lun 20/10/14	vie 24/10/14		
11	Definir tareas y estimar fechas	4 horas	lun 20/10/14	mar 21/10/14	7	Analista
12	Crear diagrama de Grantt	4 horas	mar 21/10/14	mar 21/10/14	11	Analista
13	Documentar la propuesta del proyecto	15 horas	mié 22/10/14	vie 24/10/14	12	Analista
14	Entregar la propuesta técnica	0 horas	vie 24/10/14	vie 24/10/14	13	Analista
15	Desarrollo técnico del proyecto	250 horas	lun 27/10/14	vie 09/01/15		
16	Instalar máquina virtual	5 horas	lun 27/10/14	lun 27/10/14		
17	Instalación Virtual Box	1 hora	lun 27/10/14	lun 27/10/14	14	Programador
18	Instalación distribución Linux	4 horas	lun 27/10/14	lun 27/10/14	17	Programador
19	Crear ejecutable para Windows	30 horas	mar 28/10/14	mar 04/11/14		
20	Análisis	2 horas	mar 28/10/14	mar 28/10/14	16	Analista
21	Diseño	14 horas	mar 28/10/14	vie 31/10/14	20	Diseñador
22	Personalización	2 horas	vie 31/10/14	vie 31/10/14	21	Diseñador
23	Desarrollo	10 horas	vie 31/10/14	mar 04/11/14	22	Programador
24	Pruebas	2 horas	mar 04/11/14	mar 04/11/14	23	Beta tester
25	Crear ejecutable para Mac	50 horas	mié 05/11/14	mar 18/11/14		
26	Análisis	2 horas	mié 05/11/14	mié 05/11/14	19	Analista
27	Diseño	20 horas	mié 05/11/14	mar 11/11/14	26	Diseñador
28	Personalización	5 horas	mar 11/11/14	mié 12/11/14	27	Diseñador
29	Desarrollo	18 horas	mié 12/11/14	lun 17/11/14	28	Programador
30	Pruebas	2 horas	mar 18/11/14	mar 18/11/14	29	Beta tester
31	Integrar el sistema en la plataforma Actualpacs	150 horas	mié 19/11/14	lun 05/01/15		
32	Análisis	4 horas	mié 19/11/14	mié 19/11/14	25	Analista
33	Diseño	30 horas	mié 19/11/14	jue 27/11/14	32	Diseñador
34	Desarrollo	60 horas	jue 27/11/14	mar 16/12/14	33	Programador
35	Pruebas	20 horas	mar 16/12/14	lun 22/12/14	34	Beta tester
36	Puesta en marcha	5 horas	mié 07/01/15	mié 07/01/15		
37	Implantación	5 horas	mié 07/01/15	mié 07/01/15	31	Programador;Beta tester
38	Entrega final	0 horas	mié 07/01/15	mié 07/01/15	37	Programador;Beta tester
39	Elaboración de un manual para los programadores de la empresa	10 horas	jue 08/01/15	vie 09/01/15	36	Analista;Diseñador;Programador
40	Documentación y presentación del TFG	135 horas	mié 01/04/15	jue 07/05/15		
41	Redacción de informes quincenales	10 horas	lun 13/10/14	mar 14/10/14		
42	Redacción de la memoria técnica	90 horas	mié 01/04/15	vie 24/04/15		
43	Entrega de la memoria técnica	0 horas	vie 24/04/15	vie 24/04/15	42	
44	Preparación de la presentación oral	34 horas	lun 27/04/15	mar 05/05/15	43	
45	Presentación oral	1 hora	mar 05/05/15	mar 05/05/15	44	

Figura 1. Planificación temporal de las tareas

Finalmente logré llegar a la fase de integración y puesta en marcha del sistema en la plataforma *online* Actualpacs, cumpliéndose así la planificación final.

3.2.1 Diagrama de Gantt

La Figura 2 se corresponde con el diagrama de Gantt que permite mostrar el tiempo de dedicación para las diferentes tareas a lo largo de la duración del proyecto. En el Anexo I se puede ver el diagrama junto con la planificación completa.

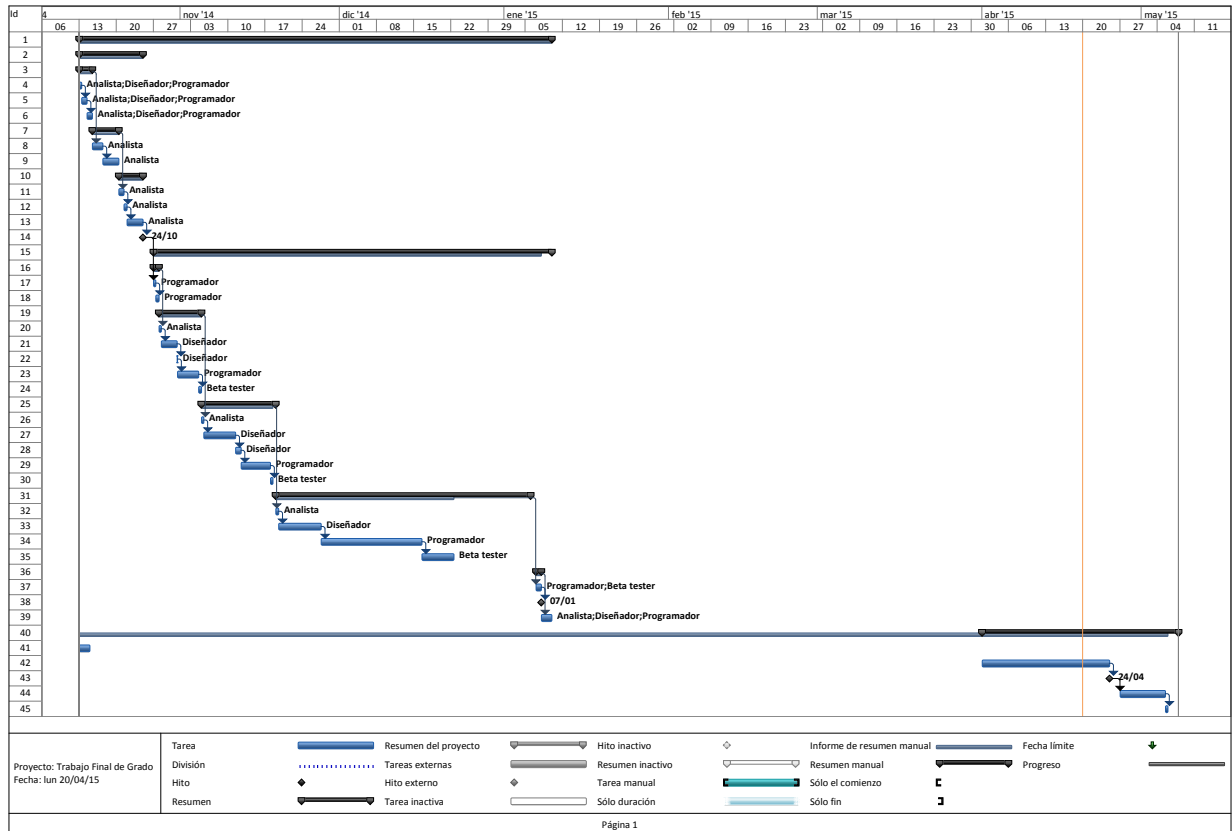


Figura 2. Diagrama de Gantt

3.2.2 Horario de trabajo

El proyecto está planteado para desarrollarlo en 300 horas, haciendo 25 horas por semana en el horario que se muestra en la Tabla 2.

Día	Lunes	Martes	Miércoles	Jueves	Viernes
Horario	09:00-14:00 15:00-18:00	11:00-14:00	09:00-13:00	09:00-14:00	09:00-14:00

Tabla 2. Horario de trabajo

3.3 Estimación de recursos del proyecto

Cuando se inicia un proyecto informático se conocen muy pocas de las funcionalidades que se deben implementar, a pesar de ello, se debe realizar una estimación de los recursos necesarios.

Los recursos se pueden clasificar en recursos Software, Hardware y Humanos. En las siguientes tablas puede observarse la clasificación hecha, así como sus correspondientes estimaciones de costes.

3.3.1 Software

Nombre del recurso	Coste
Windows 7 Profesional	120€
OS X Yosemite	0€
Virtual Box	0€
CentOS 6.5	0€
Radiant DICOM Viewer	0€
OsiriX DICOM Viewer	0€
Compresor 7zip	0€
ResEdit	0€
DMG Canvas	0€
Paquete DCMTK	0€
Sublime Text	0€
Git Extensions	0€
Git Hub private repositories	20€/mes

Tabla 3. Estimación de software

3.3.2 Hardware

Nombre del recurso	Coste
Portátil Mac Book Air	1129€
Ordenador HP Compaq	500€
Monitor LG	120€
Teclado + Ratón Logitech	25€

Tabla 4. Estimación de hardware

3.3.3 Humanos

Nombre del recurso	Tiempo en horas	Salario €/mes	€/hora	Coste
Analista	68h	2500€	15,625€	1063€
Diseñador	93h	1700€	10,625€	988€
Programador	120h	2250€	14,063€	1687€
Beta tester	29h	2100€	13,125€	380€

Tabla 5. Estimación de personal

3.3.4 Otros

Nombre del recurso	Coste
Manual Laravel: Code Bright [4]	32,58€
Certificado digital de firma de código	89€/año

Tabla 6. Estimación de otros recursos

3.3.5 Resumen

Por lo tanto, suponiendo que los distintos recursos humanos trabajan 160 horas al mes (40 horas por semana, 8 horas diarias de lunes a viernes) y que tienen una experiencia laboral de 3 años, el coste necesario para la realización del proyecto es de 6153,58€. Además, hay que tener en cuenta que cada año habría que renovar el certificado y cada mes el plan del repositorio privado. Cabe decir que la empresa ya dispone del hardware, de la licencia de Windows y del manual de Laravel, y asimismo el Mac Book Air utilizado es el mío.

Capítulo 4

Análisis y diseño del software

4.1 Análisis de requisitos

4.2 Diseño del sistema

4.3 Diseño de la interfaz

4. Análisis y diseño del sistema

Debido a que por cada una de las tres tareas principales llevadas a cabo en el desarrollo del proyecto se hace su respectivo análisis y diseño, los apartados de este capítulo harán referencia a estas tareas para un mejor entendimiento.

El diagrama de la Figura 3 pretende ilustrar cual es el flujo que seguirá la información:

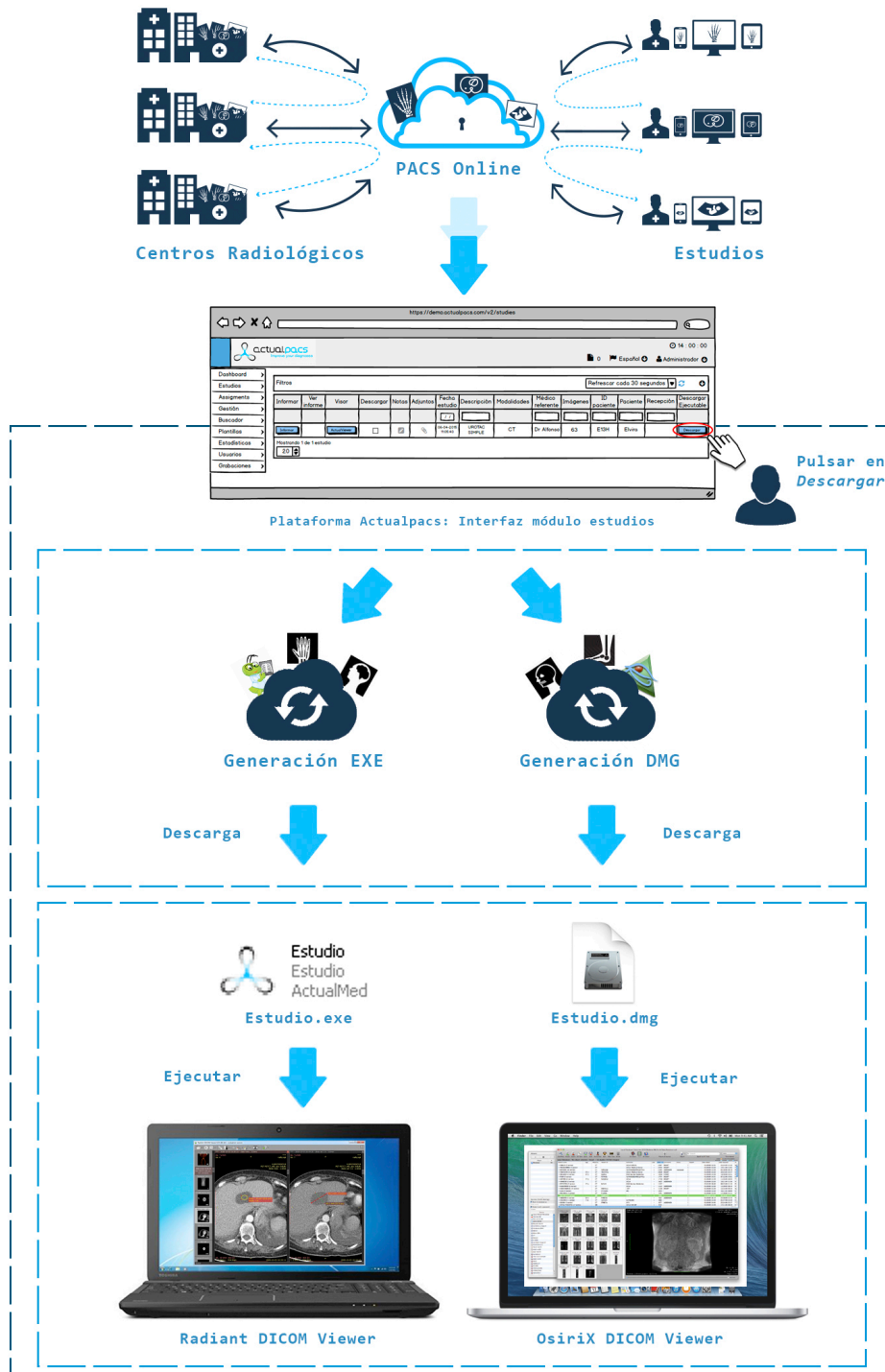


Figura 3. Flujo de la información

Los centros radiológicos realizan estudios a los pacientes y las imágenes DICOM de los estudios son archivadas en el servidor PACS. A través de la plataforma Actualpacs se puede acceder a ellos, por ejemplo para que un médico realice un diagnóstico online.

El sistema que se desarrolla en este proyecto permitirá que tanto el médico como el paciente puedan descargar un estudio de tal modo que puedan visualizar las imágenes en su ordenador, sin necesidad de estar conectados a Actualpacs, ni a internet, ni tener instalado ningún visor.

4.1 Análisis de requisitos

El propósito del siguiente apartado es definir los requerimientos que debe tener y cumplir el sistema desarrollado. Esta especificación de requisitos tiene como objetivo formalizar las funcionalidades y prerequisites de forma que haya una base con la que contrastar el desarrollo del sistema y así poder realizar el desarrollo de una forma más sencilla y guiada.

Debido a que el PACS está desplegado sobre Linux, es necesario que los ejecutables se generen a través de comandos Linux.

4.1.1 Ejecutable EXE

Se requiere que el archivo EXE contenga un visor de imágenes médicas y las imágenes del estudio del paciente. El visor elegido será el Radiant DICOM Viewer para Windows, en concreto la versión *Autorun Package for Patient CD/DVD/Flash* [5].

Esta versión está destinada para su distribución en un CD/DVD/medios Flash con un estudio DICOM. Los contenidos del archivo deben ser colocados en el directorio raíz del CD/DVD/Flash. Después de la inserción del medio, un archivo autoejecutable ejecuta automáticamente el visor, el cual realiza una búsqueda de archivos DICOM incluidos en el medio. Esta versión funciona tanto en sistemas de 32 como de 64 bits.

En nuestro caso, como queremos que el comportamiento del visor sea exactamente el mismo, es decir, que al iniciarse muestre automáticamente las imágenes del estudio que contiene, utilizaremos esta versión.

Por otro lado, se requiere empaquetar de algún modo todos estos archivos generando un EXE portable que pueda ser ejecutado sin necesidad de instalación y que al ser ejecutado se inicie el visor automáticamente.

El conocido compresor 7zip ofrece la posibilidad de conseguir esto a través de un módulo que permite crear archivos auto-extraíbles (SFX) [6].

Un archivo SFX es un fichero comprimido convertido en un ejecutable independiente que se encarga, sin necesidad de un programa compresor externo, de descomprimirse a sí mismo. Por consiguiente, comprimiremos la carpeta que contendrá el visor con las imágenes del estudio y la haremos auto-extraíble.

Se quiere que la descompresión de los archivos sea lo más transparente posible para el usuario. Con este objetivo, conviene no mostrar la barra de progreso mientras se descomprime ya que queremos ocultar al usuario que el ejecutable en realidad se trata de una carpeta comprimida que se auto-extraerá.

Por otro lado, al ejecutar un archivo EXE en Windows, si el fabricante del archivo es desconocido se muestra una advertencia de seguridad advirtiendo al usuario. Por esta razón, será necesario firmar el ejecutable para que el usuario no desconfíe de su procedencia y demostrar el origen del mismo.

4.1.2 Ejecutable DMG

En el caso de Mac también se requiere que la imagen DMG contenga un visor de imágenes médicas y las imágenes del paciente.

El visor escogido será el OsiriX DICOM Viewer [7], en concreto la versión Lite, ya que es gratuita.

A diferencia del visor Radiant que lee automáticamente las imágenes DICOM situadas en el mismo directorio raíz donde se encuentra el visor, OsiriX necesita que le indiquemos donde están situadas exactamente para poderlas abrir automáticamente.

Para ello, habrá que generar e incorporar también en el DMG un archivo denominado DICOMDIR que proporciona un índice e información de resumen para cada uno de los archivos DICOM a partir del cual el visor los lee y abre al iniciarse.

Se requiere que cuando se monte el DMG, se muestre una imagen de fondo con el logo de la clínica en el Finder y que en el centro aparezca únicamente el icono de la aplicación del visor listo para que el usuario lo pulse y se abra, ya que el usuario no tiene por qué saber qué función tiene el resto de archivos ni cuál es el que tienen que pulsar para que se abra el visor.

4.1.3 Integración del sistema en la plataforma Actualpacs

Actualpacs está desarrollado utilizando el framework Laravel para PHP. Laravel tiene como requisito tener instalado PHP y la extensión MCrypt.

También es necesario instalar un servidor web Apache y una base de datos como MySQL. Herramientas como XAMPP instalan todo esto por nosotros, por lo tanto, se decidió usar esta herramienta en el PC con Windows utilizado en el desarrollo del proyecto.

La plataforma Actualpacs está dividida en varios módulos: Estudios, Asignaciones, Gestión, Buscador, Plantillas, Estadísticas, Usuarios y Grabaciones.

Lo que se pretende con la realización de este proyecto es añadir una nueva columna al final de la tabla del módulo Estudios que muestre por cada fila de la tabla, es decir, por cada estudio, un botón con nombre “Descargar”.

Además, esta columna deberá poderse mostrar u ocultar desde el panel de configuración del administrador según las necesidades del usuario. Por defecto, no se mostrará.

Cuando el usuario pulse este botón, las imágenes asociadas al estudio seleccionado se deberán descargar desde el servidor donde están alojadas a una carpeta que estará creada previamente en Actualpacs que contendrá el visor y los archivos necesarios para la creación y firma de los ejecutables, a partir de la cual se generará el ejecutable “al vuelo” y se descargará.

Si el estudio seleccionado contiene muchas imágenes, el proceso de descarga de éstas desde el servidor y la generación del ejecutable puede tardar. Por tanto, se requiere que la plataforma dé *feedback* al usuario haciéndole ver que se está realizando la tarea después de pulsar el botón, ya que hasta que el ejecutable no empiece a descargarse no se mostrará la barra de progreso en el navegador.

Por otro lado, cuando se genere por primera vez el ejecutable de un estudio, éste no se borrará tras su descarga, así si el usuario vuelve a pulsar en el mismo estudio, ya no se tendrán que bajar las imágenes ni generarlo de nuevo, de modo que la descarga será instantánea.

4.2 Diseño del sistema

Una vez realizado el análisis y detallado los requerimientos que debe cumplir el sistema, es necesario describir cómo se van a organizar y estructurar los componentes de éste.

A modo de resumen, el diagrama de la Figura 4 muestra lo que se hará en las diferentes etapas que seguirá el sistema.

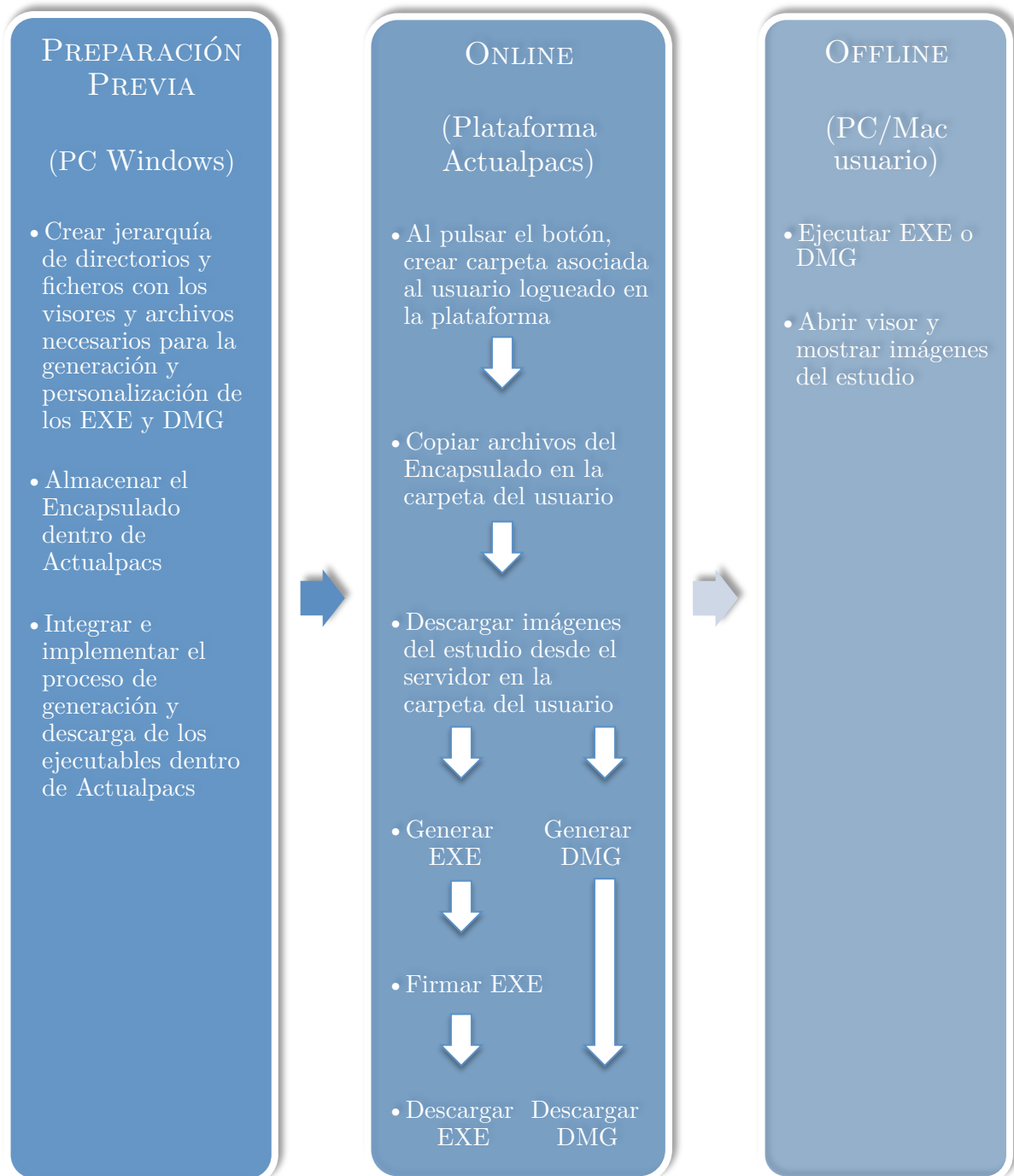


Figura 4. Etapas del sistema

4.2.1 Preparación del ejecutable EXE

En el PC con Windows crearemos una carpeta llamada *Windows* que encapsule los siguientes archivos:

- **RadiantViewer:** Carpeta que contendrá la versión portable del visor y las imágenes médicas del estudio en una subcarpeta llamada **DICOM**.
- **Modulo_SFX:** Carpeta que almacenará el archivo 7zS.sfx que contiene el módulo SFX del 7zip.
- **config.txt:** Script con información acerca del comportamiento del SFX.

De forma opcional, se puede cambiar el icono de los EXE, en concreto el archivo portable SFX. Para ello se utilizará la aplicación ResEdit [8] disponible para Windows. A continuación, se explica cómo hacerlo, aunque en el manual que se adjunta junto a la memoria se explica con todo detalle los pasos necesarios.

El primer paso que se deberá hacer es cambiar la extensión del módulo de “.sfx” a “.exe” para poder abrirlo y configurarlo. Básicamente dentro de este programa podemos elegir una imagen como icono tal y como se muestra en la Figura 5. En este caso se ha elegido la del logo de ActualMed.

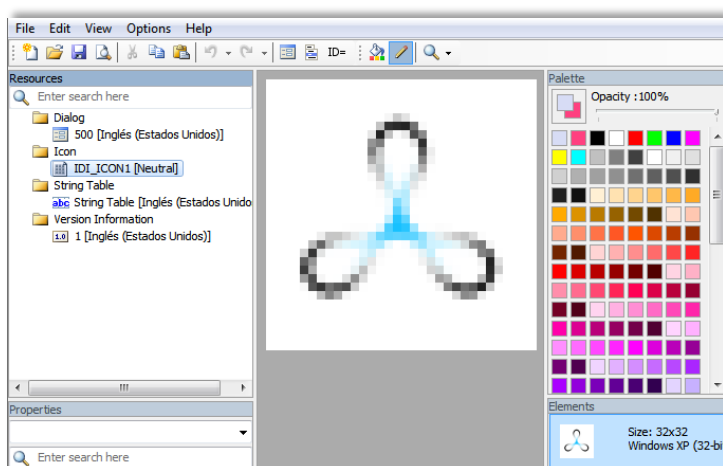


Figura 5. Cambiar icono del EXE

También se puede editar la información de la versión del ejecutable como puede verse en la Figura 6.

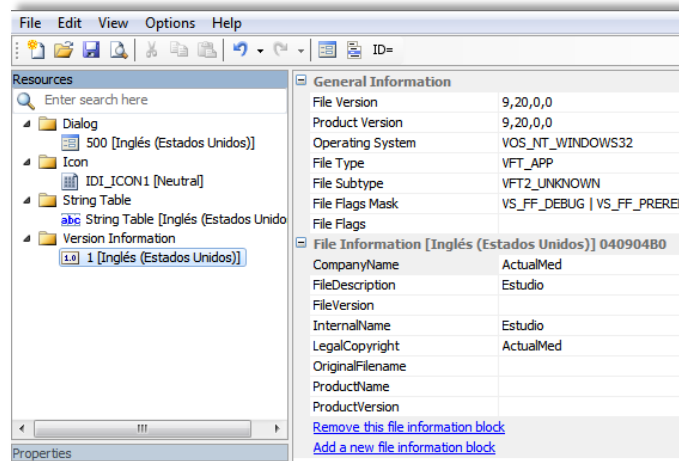


Figura 6. Editar información de la versión

Por otro lado, a partir de Windows Vista, el autor de un programa puede indicar el nivel de privilegios requerido para el sistema operativo mediante la incorporación de un archivo llamado *manifest* dentro del programa. Debido a que los módulos originales SFX del compresor 7zip no tienen este archivo, cuando se ejecuta el EXE, Windows muestra un mensaje del UAC (Control de cuentas de usuario) como el de la Figura 7.

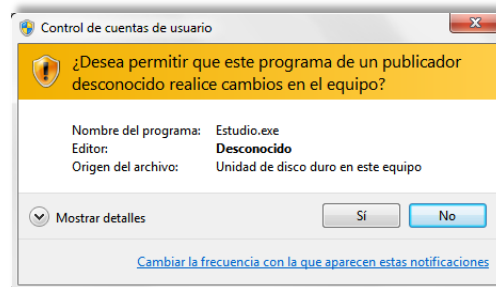


Figura 7. Control de cuentas de usuario

Podemos eliminar este aviso añadiendo un fichero *manifest* por defecto en el ResEdit como el de la Figura 8.

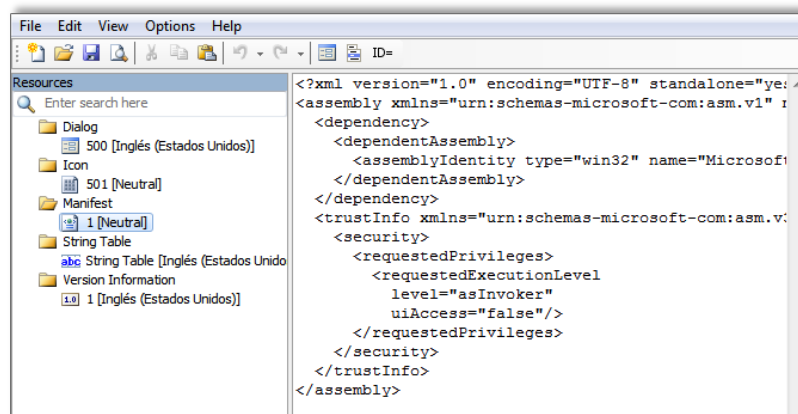


Figura 8. Archivo manifest que desactiva el UAC

Con *asInvoker* la aplicación se ejecutará con los mismos permisos que tiene el usuario. Si el usuario es administrador se ejecutará como administrador, en caso contrario no se pedirán permisos para ejecutarse.

Además, si la aplicación que se ejecuta no se instala, el PCA (Asistente para compatibilidad de programas) puede aparecer al cerrar la aplicación, alegando que no se ha podido instalar correctamente como muestra la Figura 9.

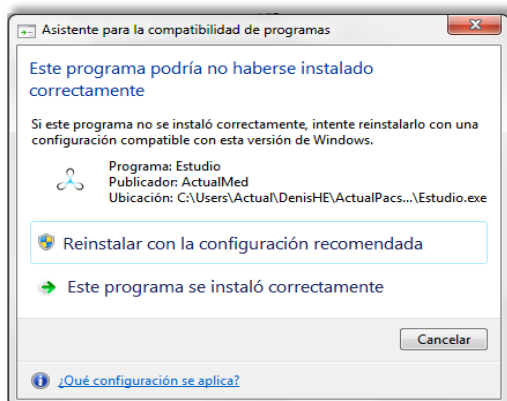


Figura 9. Asistente para compatibilidad de programas

Como en nuestro caso la aplicación que se inicia al ejecutar el EXE es el visor y no se instala sino que simplemente se abre y muestra las imágenes, habrá que desactivar este aviso ya que al usuario le saldrá cada vez que cierre el visor y puede resultarle molesto.

Para poder eliminar el aviso PCA, basta con añadir las líneas que se muestran en la Figura 10 en el archivo *manifest* anterior.

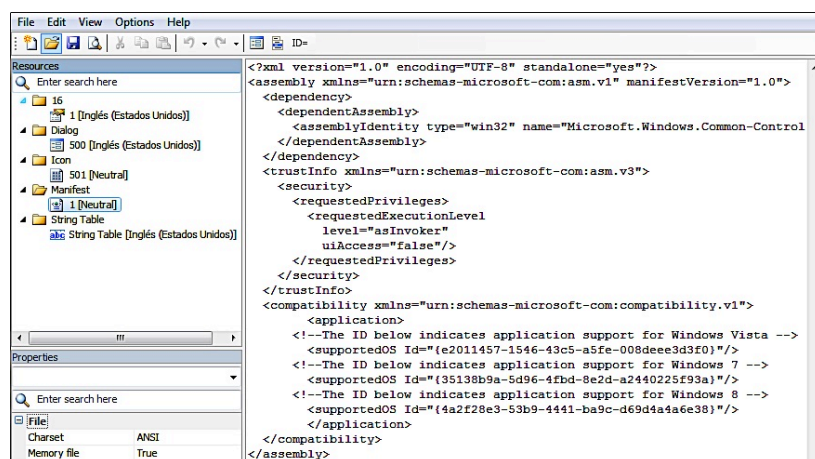


Figura 10. Archivo manifest que desactiva el UAC y PCA

Por último, una vez que se ha personalizado todo, se debe cambiar la extensión del módulo a “.sfx” de nuevo, para poder generar el ejecutable con estas configuraciones.

4.2.2 Preparación del ejecutable DMG

Del mismo modo que para el EXE, en una carpeta llamada *Mac* almacenaremos todos los archivos necesarios para generar el DMG:

- **OsiriX Launcher:** Se trata del ejecutable de la aplicación OsiriX Lite.
- **DICOM:** Carpeta que contendrá las imágenes médicas del estudio.
- **DICOMDIR:** Archivo que incluye los atributos clave de los datos de las imágenes DICOM tales como el nombre del paciente, ID del paciente, ID de estudio, fecha del estudio, etc. Y además proporciona al visor la ruta donde están situadas.

En cuanto a los archivos DMG, nos interesa que muestren una imagen de fondo al ser montados, concretamente la del logo de la clínica o centro radiológico, aunque se ha diseñado una a propósito con el logo de OsiriX para tomarla como base.

Para ello, utilizaremos el programa DMG Canvas [9] disponible para Mac. Este permite además de añadir una imagen, adaptar el tamaño de la ventana a ésta, añadir los archivos que queramos que contenga el DMG y posicionarlos a nuestro gusto.

En nuestro caso, los archivos que contendrá el DMG serán los que se han mencionado anteriormente (OsiriX Launcher, DICOM y DICOMDIR), por consiguiente, los añadiremos y distribuiremos de tal forma que el icono del visor quede centrado en medio de la ventana tal como se muestra en la Figura 11.



Figura 11. DMG Canvas

El último paso será asignar el nombre que tendrá el volumen al ser montado en el Finder de Mac, por ejemplo *Estudio* y guardar el DMG con el nombre que deseemos, por ejemplo *Estudio.dmg*.

A continuación, en Mac, montamos el DMG que hemos creado y accedemos vía terminal a `/Volumes/Estudio` y copiamos los archivos ocultos `.DS_store` y `.background` en nuestra carpeta *Mac* del Encapsulado.

Dentro de la carpeta `.background` está el archivo `backgroundImage.tiff` que ha generado el programa a partir de la imagen que le pasamos como fondo y el archivo `.DS_store` que contiene información sobre la posición de los iconos, por lo que los necesitaremos para la posterior generación de nuestros DMGs.

Por último, como al montar el DMG únicamente tiene que aparecer el icono de la aplicación OsiriX, se deberán ocultar los archivos DICOM y DICOMDIR.

En Mac existe una utilidad de línea de comandos denominada `setfile` [10] que permite hacer invisible cualquier fichero o carpeta. Para lograrlo basta con acceder a la carpeta *Mac* desde el terminal e introducir los siguientes comandos:

```
$ setfile -a V DICOM
$ setfile -a V DICOMDIR
```

El resultado se muestra en la Figura 12 donde se puede comprobar que únicamente aparece el icono del visor. Cabe decir que todo esto está también explicado paso a paso en el manual adjunto.



Figura 12. Estudio.dmg montado con los archivos DICOM y DICOMDIR ocultos

4.2.3 Integración del sistema en la plataforma Actualpacs

Para poder llevar a cabo la integración, crearemos una carpeta llamada *Encapsulado* y colocaremos todas las carpetas y archivos anteriores dentro de ésta y la almacenaremos en Actualpacs, de modo que ya esté todo preparado para generar los ejecutables a partir de los distintos archivos.

Por otro lado, dentro de Actualpacs, se deberá crear un directorio distinto por cada usuario y estudio, en el cual se descargarán las imágenes médicas del estudio seleccionado, se copiarán los archivos necesarios para generar el ejecutable desde la carpeta *Encapsulado* y se creará y alojará el archivo EXE o DMG. De este modo se evitará que se generen ejecutables corruptos si dos usuarios distintos pulsan a la vez el mismo estudio y que no se mezclen imágenes si pulsan estudios diferentes.

Asimismo, los archivos necesarios para firmar los EXE también se almacenarán en Actualpacs en una carpeta llamada *Certificado* que situaremos dentro de la carpeta *Windows*, a la cual se accederá para firmar el ejecutable antes de ser descargado.

Los archivos se organizarán del modo en el que aparecen en la Figura 13.

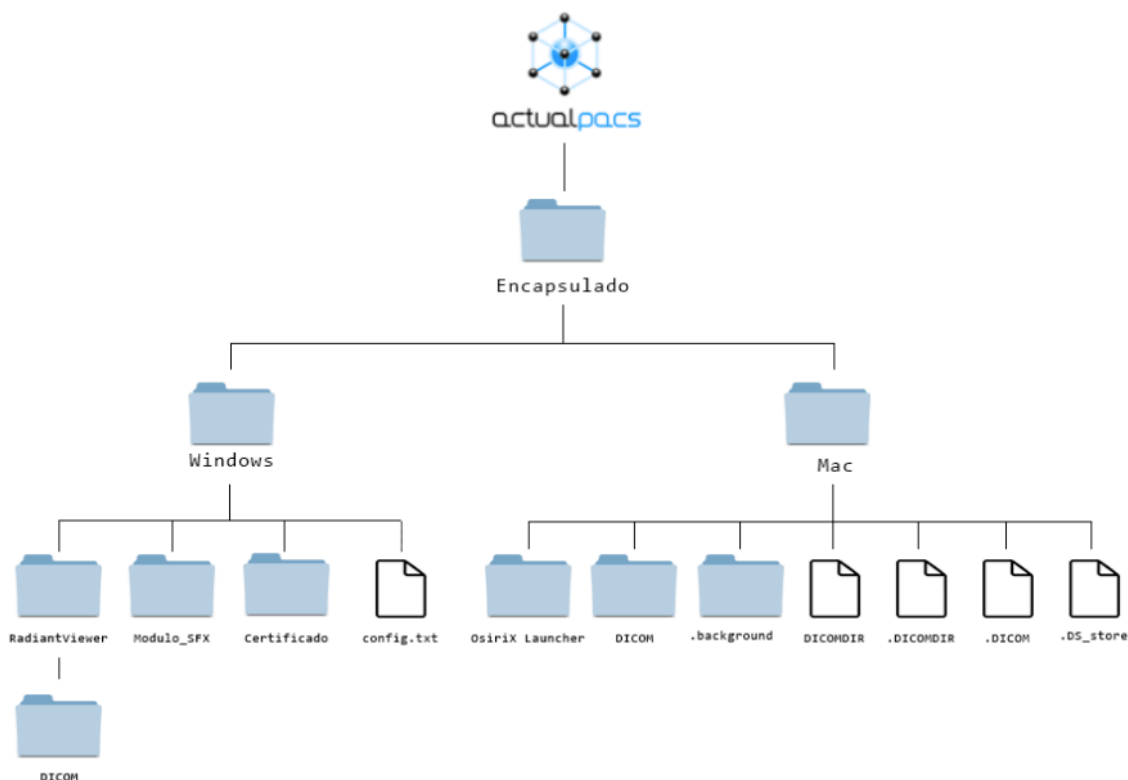


Figura 13. Jerarquía directorios Encapsulado

Capítulo 5

Implementación, pruebas y documentación

5.1 Tecnología empleada

5.2 Detalles de implementación

5.3 Validación y pruebas

5.4 Puesta en marcha

5.5 Documentación

5. Implementación, pruebas y documentación

Después de haber realizado la planificación, el análisis y el diseño, se debe realizar la implementación del proyecto. En este capítulo se detallan las distintas tecnologías y herramientas empleadas, así como los detalles de implementación, integración y validación del sistema.

5.1 Tecnología empleada

El proyecto se ha desarrollado utilizando las siguientes tecnologías y herramientas.

5.1.1 Desarrollo del proyecto

- **Microsoft Windows 7:** Sistema Operativo sobre el que se ha desarrollado el proyecto y se han personalizado y testeado los ejecutables “.exe”.
- **Mac OS X Yosemite:** Sistema Operativo sobre el cual se han personalizado y testeado los ejecutables “.dmg”.
- **CentOS 6.5:** Sistema Operativo sobre el cual se han generado los ejecutables.
- **Virtual Box:** Software de virtualización donde se ha instalado la distribución de CentOS.
- **SublimeText:** Editor de texto utilizado para la edición del código fuente. Está cargado de funcionalidades útiles y cómodas desde el punto de la usabilidad y eficiencia, convirtiendo nuestro trabajo de programación en una experiencia más sencilla y agradable.
- **Microsoft Project 2010:** Software de administración de proyectos con el que se ha realizado la planificación del proyecto y construido los diagramas.
- **Balsamiq Mockups:** Aplicación para crear maquetas para interfaces gráficas de usuario con la que se ha diseñado el prototipo de la interfaz del módulo Estudios de Actualpacs.

5.1.2 Creación y personalización del EXE

- **Radiant DICOM Viewer:** Visor de imágenes médicas para Windows. En concreto, la versión 1.9.16.7446 destinada para la distribución en un CD/DVD, compatible con Windows XP SP2 o SP3, Vista, 7 y 8.
- **7zip:** Compresor Open Source que ofrece a través de un módulo la posibilidad de crear archivos portables SFX avanzados.
- **ResEdit:** Editor de recursos para los programas de Windows. Se puede utilizar para modificar cualquiera de los recursos en los archivos EXE.
- **OpenSSL:** Consiste en un robusto paquete de herramientas de administración y bibliotecas relacionadas con la criptografía. Permite crear los archivos necesarios para poder firmar los ejecutables.
- **OSSLSigncode:** Herramienta gratuita (OpenSSL-based signcode utility) multiplataforma que utiliza OpenSSL y libcurl para firmar digitalmente archivos EXE en Linux.
- **Comodo Code Signing:** Certificado de firma de código comprado para firmar los EXE. La firma electrónica de código permite distribuir programas de forma segura garantizando la autenticidad e integridad del contenido antes de ser ejecutado y de esta forma eliminar riesgos.

5.1.3 Creación y personalización del DMG

- **OsiriX DICOM Viewer:** Visor de imágenes médicas para Mac. En concreto, la versión elegida es la 6.01 compatible con OS X 10.8, 10.9 y 10.10.
- **genisoimage:** Herramienta de línea de comandos para crear sistemas de ficheros ISO-9660/Joliet/HFS. Toma una instantánea de un árbol de directorios dado y genera una imagen binaria que corresponderá a un sistema de ficheros ISO9660 y/o HFS de Macintosh.
- **DMGCanvas:** Aplicación con la se puede crear fácilmente plantillas de diseño para las imágenes de disco DMG. Crearlas es fácil gracias a su editor gráfico integrado. Se puede escoger un color de fondo o una imagen, luego personalizar el icono de la imagen de disco, su posición y acceder a una multitud de ajustes.

- **Paquete DCMTK:** DCMTK incluye una colección de bibliotecas y aplicaciones para examinar, construir y convertir archivos de imágenes de DICOM. Se utiliza para crear los archivos DICOMDIR.

5.1.4 Integración en Actualpacs

- **Laravel 4:** Framework PHP con el que está implementado ActualPacs. Permite el uso de una sintaxis elegante y expresiva para crear código de forma sencilla y permitiendo multitud de funcionalidades. Intenta aprovechar lo mejor de otros frameworks y las características de las últimas versiones de PHP. Está basado en el patrón Modelo-Vista-Controlador.
- **PHP5:** Como lenguaje web debido a su gran comunidad, rendimiento, portabilidad y por ser una exigencia del propio framework PHP utilizado. En concreto se utiliza la versión 5.4 y la extensión Mcrypt.
- **MySQL 5:** Sistema de gestión de base de datos. Debido a su facilidad de uso y compatibilidad con Apache gracias a herramientas como WAMP. Mediante la herramienta PHPMyAdmin, disponible en el paquete XAMPP, es posible administrar la base de datos debido a que ofrece utilidades para la creación, modificación y eliminación de base de datos, así como de las tablas y las relaciones.
- **HTML5:** Lenguaje básico de la World Wide Web, por la cantidad de nuevos elementos y atributos que incorpora. Así como por la simplificación y compatibilidad que ofrece.
- **JQuery:** Librería para JavaScript por ser soportado por multitud de navegadores y porque ofrece un conjunto de funciones que permiten la manipulación fácil y eficaz del DOM, manejo sencillo de eventos y posee funciones Ajax cómodas de manejar.
- **Notify.js:** Plugin de jQuery que proporciona notificaciones y que se utiliza para proporcionar *feedback* al usuario.
- **Apache:** Servidor web mediante el cual se proporciona acceso a la base de datos y a las funciones PHP.
- **GitHub** junto con **GitExtensions** para el control de versiones y el almacenado del proyecto en un repositorio remoto. Estas herramientas facilitan el control, restauración y trabajo colaborativo en el proyecto.

5.2 Detalles de implementación

En este apartado se explica cómo crear el sistema que genera un ejecutable tanto para Windows como para Mac en una máquina virtual con CentOS y cómo integrarlo dentro de la plataforma Actualpacs.

5.2.1 Ejecutable EXE

Para la creación del fichero auto-extraíble SFX se necesitan tres archivos:

- **Estudio.7z**: Archivo comprimido con el contenido de la carpeta *RadiantViewer*.
- **7zS.sfx**: Módulo SFX del 7zip.
- **config.txt**: Script con el que se le indica al módulo SFX qué programa deberá abrir una vez se extraiga. En nuestro caso queremos que se abra el visor nada más ejecutar el EXE por lo que este será su contenido:

```
1 ;!@Install@!UTF-8!  
2 RunProgram="RadiantViewer\RA32\viewer32.exe"  
3 Progress="no"  
4 ;!@Install@!UTF-8!
```

- RunProgram: Ruta del programa que se abrirá al ejecutar el EXE.
- Progress: “yes” o “no”. Esto muestra u oculta la barra de progreso mientras se está descomprimiendo el archivo. En esta ocasión no se desea que se muestre.

En primer lugar, se instalará el compresor 7zip:

```
$ yum install p7zip
```

A continuación, se deberá comprimir la carpeta que contiene el visor y darle un nombre, por ejemplo *Estudio.7z*:

```
$ 7za a Estudio.7z RadiantViewer
```

Seguidamente, habrá que hacer una copia binaria de los tres archivos creando el SFX auto-extraíble *Estudio.exe* final, listo para ser ejecutado en cualquier PC con Windows:

```
$ cat Modulo_SFX/7zS.sfx config.txt Estudio.7z > Estudio.exe
```

Y por último, se tendrá que borrar la carpeta comprimida dado que no se va a necesitar más:

```
$ rm -rf Estudio.7z
```

Es importante señalar que al ejecutar el EXE, el módulo extrae el contenido del auto-extraíble en la carpeta temporal del sistema y elimina los archivos temporales después de que la aplicación finalice, es decir, al cerrar el visor.

5.2.1.1 Pruebas unitarias ejecutable EXE

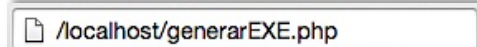
Para probar el correcto funcionamiento antes de integrarlo en la plataforma, en CentOS instalamos PHP y Apache, copiamos dentro del directorio `/var/www/html` la parte de *Windows* del Encapsulado y damos permiso a todos los ficheros con:

```
$ chown -R apache:apache Encapsulado
```

Después, creamos el archivo `generarEXE.php` que mediante funciones `exec()` ejecutará los comandos anteriores y con `readfile()` descargará el ejecutable:

```
01. <?php
02. //-----GENERAR EXE-----
03. exec('7za a ./Encapsulado/Windows/Estudio.7z ./Encapsulado/Windows/RadiantViewer');
04. exec('cat ./Encapsulado/Windows/Modulo_SFX/7zS.sfx ./Encapsulado/Windows/config.txt
05. ./Encapsulado/Windows/Estudio.7z > ./Encapsulado/Windows/Estudio.exe');
06. exec('rm -r ./Encapsulado/Windows/Estudio.7z');
07.
08. //-----DESCARGAR EXE-----
09. $file = './Encapsulado/Windows/Estudio.exe';
10. if(file_exists($file)){
11.     header('Content-Description: File Transfer');
12.     header('Content-Type: application/octet-stream');
13.     header('Content-Disposition: attachment; filename='.basename($file));
14.     header('Content-Length: '.filesize($file));
15.     readfile($file);
16.     exec('rm ./Encapsulado/Windows/Estudio.exe');
17. }else{
18.     echo "El fichero $file no existe";
19. }
20. ?>
```

De este modo, se puede probar a generar y descargar el ejecutable a través del navegador poniendo como URL la ruta local al archivo:



`/localhost/generarEXE.php`

5.2.1.2 Certificado de firma de código digital

La Figura 16 se corresponde con la advertencia de seguridad que muestra Windows al ejecutar el EXE generado ya que no está firmado y el fabricante del archivo es desconocido.

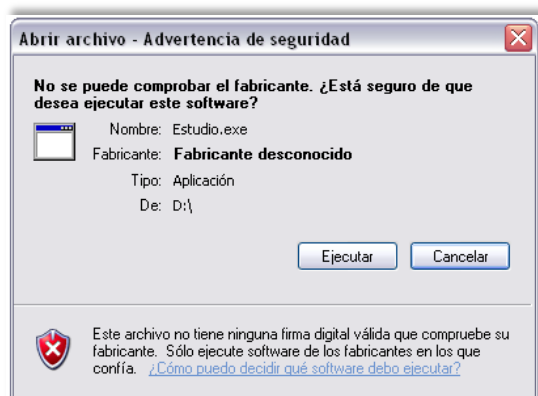


Figura 16. Editor del ejecutable desconocido

Lo primero que se debe hacer es comprar un certificado de firma de código digital e instalarlo en el navegador del PC con Windows. En nuestro caso elegimos el *Comodo Code Signing* de SSLPoint [11] ya que es el más económico que encontramos.

Para poder usar nuestro certificado para firmar el ejecutable en Linux se utilizará la herramienta *OSSLSigncode*.

Para ello, necesitaremos exportarlo desde el navegador en formato PFX (Personal Information Exchange) y convertirlo en los archivos PVK (Private Key) y SPC (Software Publishing Certificate) [12] que se utilizarán para firmar el ejecutable.

Para exportar la clave privada “.pvk” y el certificado de publicación de software “.spc” desde el archivo “.pfx”, se necesitarán las herramientas de *OpenSSL*.

El formato PKCS#12 o PFX es un formato binario que almacena certificados y la clave privada de forma encriptada. Suelen ser utilizados en Windows para importar y exportar certificados y claves privadas. Si convertimos un fichero “.pfx” a formato PEM utilizando *OpenSSL* todos los certificados y la clave privada serán almacenados en un único fichero.

Los pasos a seguir son los siguientes [13]:

1. Exportar el certificado desde Firefox en formato PFX:

- En Firefox, accedemos a *Herramientas* y seleccionamos *Opciones*.
- En la sección *Avanzado*, pestaña *Cifrado*, pulsamos *Ver certificados*.
- En la pestaña *Sus certificados*, seleccionamos el certificado a exportar y pulsamos en el botón *Copia de seguridad*.
- Pondremos un nombre al archivo de la copia y seleccionaremos la ubicación donde deseamos almacenarlo.
- Definiremos una contraseña de respaldo que protegerá el fichero.
- Si todo se ha realizado correctamente, aparecerá un mensaje indicado que la exportación se ha realizado con éxito.

2. Instalar las herramientas *OpenSSL* y *osslsigncode* en CentOS:

```
$ yum install openssl  
$ yum install osslsigncode
```

3. Extraer la clave privada del archivo “.pfx” a formato PEM:

```
$ openssl pcks12 -in ActualTecInnovaciónTecnológica,S.L.pfx  
-nocerts -nodes -out key.pem
```

Se solicitará que introduzcamos la contraseña del PFX. Esta es la contraseña que le dimos al archivo al exportarlo.

4. Convertir la clave privada PEM a formato PVK:

```
$ openssl rsa -in key.pem -outform PVK -pvk-strong -out  
ActualTecInnovaciónTecnológica,S.L.pvk
```

5. Extraer nuestro certificado del archivo “.pfx”:

```
$ openssl pcks12 -in ActualTecInnovaciónTecnológica,S.L.pfx  
-nokeys -nodes -out cert.pem
```

Habrá que introducir de nuevo la contraseña del PFX.

6. Convertir el certificado PEM a formato SPC:

```
$ openssl crl2pkcs7 -nocrl -certfile cert.pem -outform DER -out  
ActualTecInnovaciónTecnológica,S.L.spc
```

7. Por último, firmaremos el EXE con la herramienta *osslsigncode*:

```
$ osslsigncode -spc ActualTecInnovaciónTecnológica,S.L.spc -key  
ActualTecInnovaciónTecnológica,S.L.pvk -in Estudio.exe -out  
Estudio-Signed.exe
```

En consecuencia, si se vuelve a ejecutar el EXE, Windows ya no muestra la misma advertencia de seguridad, sino una nueva señalando que el archivo procede de internet y aconsejando al usuario que únicamente ejecute software de editores en los que confía, como podemos ver en la Figura 17. Como esta vez el editor es de confianza y no uno desconocido, el usuario lo podrá ejecutar sin desconfiar de su procedencia.

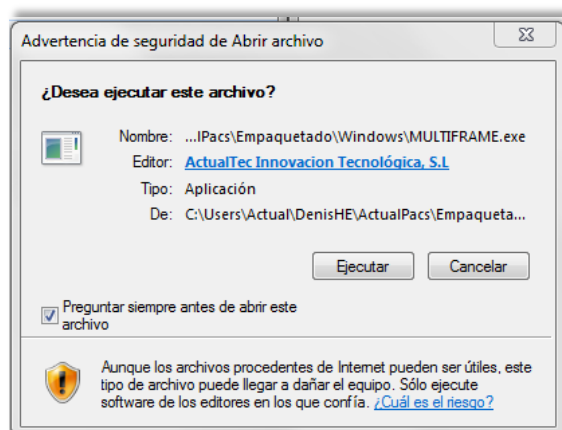


Figura 17. Editor del ejecutable de confianza

5.2.2 Ejecutable DMG

Para crear la imagen DMG, se empleará la herramienta de línea de comandos *genisoimage*. Ésta permite crear una imagen “.dmg” a partir de un directorio.

La sintaxis del comando es la siguiente [14]:

```
$ genisoimage -quiet -V <Volumen> -R -apple -o <archivo.dmg> <D>
```

- -quiet: No se proporcionará una salida con el progreso
- -V: Especifica la etiqueta del volumen
- -R -apple: Crea una imagen DMG
- -o: Nombre del archivo que se va a generar
- <D>: Directorio origen

En primer lugar, se instalará la herramienta en CentOS:

```
$ yum install genisoimage
```

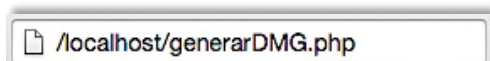
En segundo lugar, generamos el DMG a partir de la carpeta *Mac* que contiene los archivos mencionados en el capítulo anterior:

```
$ genisoimage -quiet -V Estudio -R -apple -o Estudio.dmg /Encapsulado/Mac
```

5.2.2.1 Pruebas unitarias ejecutable DMG

En último lugar, copiamos dentro del directorio `/var/www/html` la carpeta *Mac* y creamos el fichero `generarDMG.php` para probar también a generar y descargar el ejecutable a través del navegador:

```
01. <?php
02. //-----GENERAR DMG -----
03. exec('genisoimage -quiet -V Estudio -R -apple -o ./Encapsulado/Estudio.dmg ./Encapsulado/Mac/');
04.
05. //----- DESCARGAR DMG -----
06. $file = './Encapsulado/Mac/Estudio.dmg';
07. if(file_exists($file)){
08.     header('Content-Description: File Transfer');
09.     header('Content-Type: application/octet-stream');
10.     header('Content-Disposition: attachment; filename='.basename($file));
11.     header('Content-Length: '.filesize($file));
12.     readfile($file);
13.     exec('rm ./Encapsulado/Mac/Estudio.dmg');
14. }else{
15.     echo "El fichero $file no existe";
16. }
17. ?>
```



5.2.3 Integración del sistema en la plataforma Actualpacs

Llegados a este punto, hay que integrar el proceso de generación y descarga de los distintos ejecutables en Actualpacs y adaptarlo al framework Laravel.

La Figura 18 pretende sintetizar de un modo general, los principales pasos llevados a cabo para conseguirlo.

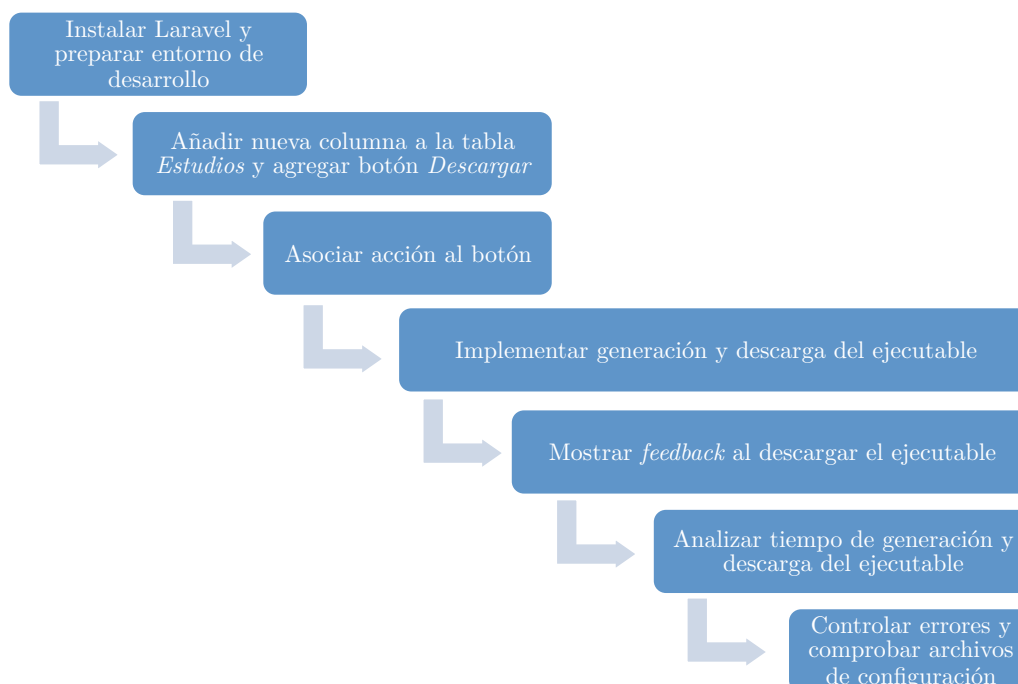


Figura 18. Pasos de la integración

5.2.3.1 Instalar Laravel y preparar entorno de desarrollo

Para instalar Laravel hace falta un manejador de paquetes y dependencias entre paquetes para PHP llamado Composer. También hará falta instalar una consola de GIT [15]. En Windows la instalación se puede hacer mediante un instalador ejecutable que se descarga de la página oficial de Composer [16].

Al ejecutarlo, se nos solicita la ubicación de nuestro php.exe que será C:\xampp\php\php.exe. El instalador se encargará de modificar la variable PATH para que podamos hacer uso de Composer desde cualquier lugar en la consola.

Una vez instalado Composer, usando la consola GIT, hay que ir a la carpeta que utiliza nuestro servidor C:\xampp\htdocs y ejecutar el siguiente comando:

```
C:\> composer create-project laravel/laravel -prefer-dist
```

Composer se encargará de descargar el proyecto base de Laravel, el framework y todas sus dependencias.

A continuación, clonamos el proyecto de pruebas *actualpacsv2* de la aplicación Actualpacs desde GitHub en C:\xampp\htdocs con Git Extensions.

Después, hay que crear la base de datos *laravel* que utiliza algunos estudios médicos de ejemplo que están alojados en un servidor PACS de la empresa.

Usando de nuevo la consola GIT, hay que ir a la carpeta del proyecto y ejecutar los siguientes archivos:

```
C:\xampp\htdocs\actualpacsv2> migrations.bat  
C:\xampp\htdocs\actualpacsv2> seeders.bat
```

- migrations.bat: Crea todas las tablas de la base de datos que tiene el proyecto *actualpacsv2*.
- seeders.bat: Crea dos usuarios de ejemplo para poder trabajar (admin y superadmin).

Seguidamente, habrá que insertar la carpeta *Encapsulado* que contiene las subcarpetas *mac* y *windows*, dentro del directorio *app* del proyecto *actualpacsv2*.

Una vez realizados estos pasos previos, ya tenemos todo listo para llevar a cabo la integración.

Para el desarrollo del código fuente empleamos el editor de texto SublimeText [17]. En la Figura 19 se muestra la jerarquía de directorios al abrir el proyecto de pruebas *actualpacs2* en el editor.

Las principales carpetas son:

- /app: Contiene los controladores, modelos, vistas y configuraciones de la aplicación.
- /public: Todas las peticiones y solicitudes a la aplicación pasan por esta carpeta, ya que en ella se encuentra el `index.php`, este archivo es el que inicia todo el proceso de ejecución del framework.
- /vendor: Aloja todas las librerías que conforman el framework y sus dependencias
- /lang: En esta carpeta se guardan archivos PHP que contienen arrays con los textos de diferentes lenguajes.
- /app/config: Aquí se pueden encontrar todos los archivos de configuración de la aplicación como base de datos, cache, correos, sesiones, etc.
- /app/controller: Archivos con las clases de los controladores que sirven para interactuar con los modelos, las vistas y manejar la lógica de la aplicación.
- /app/model: Los modelos son clases que representan la información (datos) de la aplicación y las reglas para manipularlos. En la mayoría de los casos cada tabla de la base datos esta representada con un modelo.
- /app/views: Este directorio contiene las plantillas de HTML que usan los controladores para mostrar la información.
- /app/modules: En el caso de Actualpacs, la aplicación está desglosada en varios módulos que a su vez contienen los controladores, modelos, vistas y configuraciones de cada módulo.
- /app/encapsulated: Este directorio contiene los subdirectorios *windows* y *mac* que almacenan los ficheros necesarios para la generación y personalización de los distintos ejecutables.

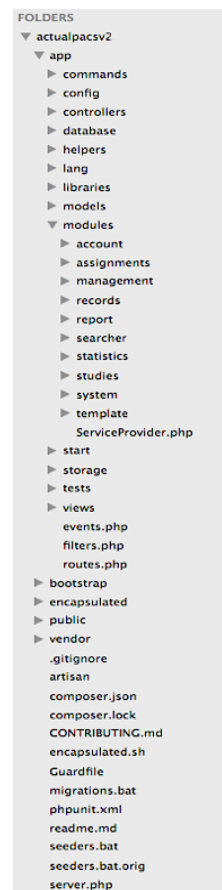


Figura 19. Jerarquía directorios *actualpacs2*

5.2.3.2 Añadir nueva columna y botón Descargar

Lo primero que habrá que hacer es añadir la nueva columna en la tabla que muestra los estudios del módulo Estudios, que contenga por cada fila un botón con nombre “Descargar”. Los pasos a seguir son los siguientes:

- Añadir en el fichero `/app/modules/studies/routes.php` la siguiente línea:

```
01. Route::post('studies/downloadExecutable', 'StudiesController@downloadExecutable');
```

- En `/models/Pacsdb/Study.php` añadimos el *case* `downloadExecutable` y creamos el botón “Descargar”.

```
01. case 'downloadExecutable':
02.     $out = '<a href="studies/downloadExecutable?study_iuid='. $item['study_iuid'].
03.           '&cname='. $item['pacsconnection'] . '&description='. $item['study_desc']. "'>
04.           <button class="btn btn-info download_executable">Descargar</button></a>';
05.     break;
```

- En `models/Pacsdb/Fields.php` añadimos el *case* `downloadExecutable`.
- En `/modules/studies/database` creamos el fichero `AddColumnToStudiesColumnsTableSeeder.php`.

Para añadir la columna `downloadExecutable` a la base de datos ejecutamos en la consola:

```
C:\xampp\htdocs\actualpacs2> php artisan db:seed
--class=AddColumnToStudiesColumnsTableSeeder
```

- En `/modules/studies/database/seeds/StudiesSeeder.php` añadir al final de `$seeders` la línea `AddColumnToStudiesColumnsTableSeeder`.
- En `/modules/studies/lang/es/studies.php` añadimos la traducción:

```
01. 'downloadExecutable' -> 'Descargar'
```

- Por último, en `/modules/studies/lang/en/studies.php` añadimos:

```
01. 'downloadExecutable' -> 'Download'
```

La nueva columna se podrá mostrar u ocultar desde el panel de configuración del usuario, dependiendo de sus necesidades.

El código fuente de los distintos ficheros mencionados puede verse en el anexoII.

5.2.3.3 Asociar al botón la acción de descargar ejecutable

El segundo paso será asociarle una acción al botón. Cuando se pulse se llamará a la función *downloadExecutable()* que crearemos en el fichero *StudiesController.php* correspondiente al controlador del módulo Estudios.

La función completa puede verse en el anexo II. A continuación, se muestra desglosada en tres partes:

```
01. public function downloadExecutable(){
02.
03.     function seoUrl($string) {
04.
05.         if($string == "")
06.             $string = "Study";
07.         #Remove all non-alphanumeric characters from data
08.         $string = preg_replace("/[^A-Za-z0-9]/", "", $string);
09.         #Replace space with _
10.         $string = str_replace(" ", "_", $string);
11.
12.         return $string;
13.     }
14.
15.     $study_uid = Input::get('study_uid');
16.     $cname = Input::get("cname");
17.     $entity = Connection::where('name', '=', $cname)->first()->toArray();
18.     $study_description = Input::get('description');
19.     $study_description = seoUrl($study_description);
20.     $session_path = public_path() . DIRECTORY_SEPARATOR . 'temp' . DIRECTORY_SEPARATOR . Session::get('user.id');
21.
22.     $user_agent = $_SERVER['HTTP_USER_AGENT'];
23.     // Windows
24.     $win = strpos($user_agent, 'Windows') ? true : false;
25.     // Macintosh
26.     $mac = strpos($user_agent, 'Macintosh') ? true : false;
```

La función interna *seoUrl(\$string)* tiene como finalidad poner el nombre de la descripción del estudio al ejecutable, por ejemplo RXRODILLA.exe.

Recibe la descripción del estudio y elimina todos los caracteres que pueda contener y si contiene algún espacio lo cambia por barra baja “_”. En el caso de no tener descripción, se le asignará como tal la cadena “Study”.

Las líneas posteriores obtienen la descripción del estudio seleccionado y en la variable *\$session_path* se guarda la ruta de la carpeta temporal *temp* correspondiente al usuario que ha ingresado en Actualpacs (public/temp/usr.id), ya que se creará un directorio distinto por cada usuario y estudio.

Haciendo uso del método *\$_SERVER['HTTP_USER_AGENT']* recogemos el contenido de la cabecera *user-agent* para poder averiguar con que sistema operativo está accediendo el usuario a la plataforma.

Después, se creará una nueva instancia de la clase *CreateExecutable* y se accederá a sus métodos utilizando el operador de flecha (->).

```

28.     if($user_agent){
29.
30.         $createExecutable = new CreateExecutable($study_iuid);
31.
32.         if($win){
33.
34.             $error = false;
35.             $exe_path = $session_path . DIRECTORY_SEPARATOR . 'studyDownloadExe' . DIRECTORY_SEPARATOR . $study_iuid . DIRECTORY_SEPARATOR . $study_description . '.exe';
36.             if(!file_exists($exe_path)){
37.                 $study_path = $session_path . DIRECTORY_SEPARATOR . 'studyDownloadExe' . DIRECTORY_SEPARATOR . $study_iuid . DIRECTORY_SEPARATOR;
38.                 if(!file_exists($study_path.'RadiantViewerLite')){
39.                     mkdir($study_path.'RadiantViewerLite', 0755, true);
40.                     $radiant_path = DIRECTORY_SEPARATOR . 'var' . DIRECTORY_SEPARATOR . 'www' . DIRECTORY_SEPARATOR . 'encapsulated' . DIRECTORY_SEPARATOR .
41.                         'windows' . DIRECTORY_SEPARATOR . 'RadiantViewer' . DIRECTORY_SEPARATOR;
42.
43.                     File::copyDirectory($radiant_path, $study_path . 'RadiantViewerLite');
44.
45.                     //Download DICOM
46.                     $result = '';
47.                     $ini = microtime(true);
48.                     $command_result = DicomCommands::downloadStudy($name, "yes", $study_iuid, $entity['aetitle'], $entity['viewerhost'], $entity['viewerport'],
49.                         $study_path.'RadiantViewerLite' . DIRECTORY_SEPARATOR . 'DICOM');
50.                     $fin = microtime(true);
51.                     System::log('study_download_time_win', '', $fin-$ini);
52.                 }
53.
54.                 $ini = microtime(true);
55.                 $error = $createExecutable->generateEXE($study_description);
56.                 $fin = microtime(true);
57.                 System::log('create_exe_time', '', $fin-$ini);
58.             }
59.             if(!$error){
60.                 return $createExecutable->downloadEXE($study_description, $study_iuid);
61.             }
62.         }

```

Por un lado, si se ha accedido desde Windows, en la variable *\$exe_path* se guardará la ruta donde se almacenará el archivo EXE y en *\$study_path* la ruta donde se descargarán desde el servidor las imágenes del estudio.

En la ruta *\$study_path* se creará la carpeta *RadiantViewerLite* y dentro se copiarán los archivos que contiene la carpeta *RadiantViewer* (*\$radiant_path*) que hay en el interior del Encapsulado con la función *File::copyDirectory*.

Con *DicomCommands::downloadStudy* se descargarán las imágenes del estudio dentro de la carpeta /RadiandViewerLite/DICOM.

La función *downloadStudy* se ha creado con la ayuda de uno de los programadores de la empresa en el fichero /app/libraries/DicomCommands.php:

```

01.     public static function downloadStudy($conns, $compression, $study_iuid, $aetitle, $host, $port, $study_path)
02.     {
03.         $default = ini_get('max_execution_time');
04.         set_time_limit(0);
05.         $wadoData = Study::getStudyWadoData($conns, $study_iuid);
06.         foreach ($wadoData as $key => $data) {
07.             $wadoUrl = self::buildWadoUrl($compression, $study_iuid, $data['series_iuid'], $data['sop_iuid'], $host, $port);
08.             $image = self::getImageFromWado($wadoUrl);
09.             self::saveImageTo($image, $data['sop_iuid'], $study_path);
10.         }
11.     }

```

Esta función se conecta con el servidor y descarga las imágenes del estudio seleccionado en la ruta que se le pasa como último parámetro. Del mismo modo se le puede pasar como segundo parámetro “yes” o “no” para indicar si se desea que las imágenes se descarguen comprimidas o no desde el servidor.

Después, se llama a la función *generateEXE* de la clase *createExecutable* pasándole la descripción del estudio y, si no ha habido ningún error al generarlo, se llama a la función *downloadEXE* de la clase *createExecutable* pasándole también la descripción del estudio y su identificador.

Por otra parte, si se ha accedido desde un Mac, en la variable `$dmg_path` se guardará la ruta donde se almacenará la imagen DMG y en `$study_path` la ruta donde se descargarán las imágenes del estudio.

En la ruta `$study_path` se creará la carpeta `OsiriXLite` y dentro se copiarán los archivos que contiene la carpeta `mac` (`$mac_path`) que hay en el interior del Encapsulado.

```
74.         elseif ($mac) {
75.
76.             $error = false;
77.
78.             $dmg_path = $session_path . DIRECTORY_SEPARATOR . 'studyDownloadDmg' . DIRECTORY_SEPARATOR .
79.             $study_uid . DIRECTORY_SEPARATOR . $study_description.'.dmg';
80.
81.             if(!file_exists($dmg_path)){
82.
83.                 $study_path = $session_path . DIRECTORY_SEPARATOR . 'studyDownloadDmg' . DIRECTORY_SEPARATOR .
84.                 $study_uid . DIRECTORY_SEPARATOR;
85.
86.                 if(!file_exists($study_path.'OsiriXLite')){
87.                     mkdir($study_path.'OsiriXLite', 0755, true);
88.
89.                     $mac_path = DIRECTORY_SEPARATOR . 'var' . DIRECTORY_SEPARATOR . 'www' . DIRECTORY_SEPARATOR .
90.                     'encapsulated' . DIRECTORY_SEPARATOR . 'mac' . DIRECTORY_SEPARATOR;
91.
92.                     File::copyDirectory($mac_path, $study_path.'OsiriXLite');
93.                     exec('chmod 777 -R '.$study_path);
94.
95.                     //Download DICOM
96.                     $result = '';
97.                     $ini = microtime(true);
98.
99.                     $command_result = DicomCommands::downloadStudy($cname, "yes", $study_uid, $entity['aetitle'],
100.                    $entity['viewerhost'], $entity['viewerport'], $study_path.'OsiriXLite' . DIRECTORY_SEPARATOR . 'DICOM');
101.
102.                     $fin = microtime(true);
103.                     System::log('study_download_time_mac', '', $fin-$ini);
104.                 }
105.
106.                 $ini = microtime(true);
107.                 $error = $createExecutable->generateDMG($study_description);
108.                 $fin = microtime(true);
109.                 System::log('create_dmg_time', '', $fin-$ini);
110.             }
111.
112.             if(!$error){
113.                 return $createExecutable->downloadDMG($study_description, $study_uid);
114.             }
115.         }
116.     }
117.     return '';
118. }
```

De nuevo, con `DicomCommands::downloadStudy` descargaremos las imágenes del estudio seleccionado en `/OsiriXLite/DICOM`.

Finalmente, se llama a la función `generateDMG` de la clase `createExecutable` para que genere la imagen DMG y, si no ha habido ningún error, se llama a la función `downloadDMG` para que la descargue.

5.2.3.4 Analizar tiempo de generación y descarga del ejecutable

La función `microtime()` devuelve la fecha Unix actual en microsegundos:

```
01. $ini = microtime(true);
02. $fin = microtime(true);
03. System::log('...', '', $fin-$ini);
```

Éstas líneas se han utilizado para medir los tiempos de descarga de las imágenes DICOM desde el servidor y los de generación y descarga de los ejecutables, para así reducirlos lo máximo posible.

Estos tiempos se insertan en la base de datos *laravel* en una tabla creada expresamente para los *logs*, con un identificador para poder distinguir fácilmente cada tiempo a que tarea se corresponde y a la cual se accedía a medida que se implementaba el código para ver que parte tardaba más e intentar agilizarla. Gracias a esto se comprobó que lo que más tiempo conlleva es descargar las imágenes.

Inicialmente se pensó en mantener automáticamente, cada cierto tiempo, una copia local de las imágenes de cada estudio para que así se pudiera generar el ejecutable rápidamente a partir de éstas y no esperar a que se descarguen antes. Pero al final, se ha decidido no hacerlo ya que si indicamos que las imágenes se bajen comprimidas desde el servidor, se reduce significativamente el tiempo y se ha considerado aceptable.

Además, cuando se genera por primera vez el ejecutable de un estudio, éste no se borra tras su descarga de modo que si se descarga de nuevo el mismo estudio, ya no se tendrán que bajar las imágenes ni generarlo, por lo que la descarga es instantánea.

Por otro lado, el tiempo de descarga del ejecutable dependerá del ancho de banda del que disponga cada usuario.

Lo que si que se ha podido hacer es agilizar es el proceso de generación del ejecutable de Windows empaquetando los archivos necesarios para generarlo, en lugar de comprimirlos, con la opción “*-mx=0*” del 7zip.

5.2.3.5 Mostrar notificación al descargar

Para que el usuario no crea que no ha ocurrido nada al pulsar el botón de “Descargar” hasta que empieza a descargarse el ejecutable y el navegador muestra la barra de progreso, se utiliza el plugin *Notify.js* [18] de jQuery para mostrar la notificación de “Descargando” en la zona superior de la página mientras se descargan las imágenes desde el servidor y se genera el ejecutable. Para ello, en el fichero existente *studies.index.js* añadimos lo siguiente:

```
01. $('table#table_studies').on('click', '.download_executable', function(e)
02. {
03.     e.preventDefault();
04.     var href = $(this).parent('a').attr('href');
05.     $.noty.closeAll();
06.     noty({text: '<br><i>'+message_downloading+'</i>', type: 'alert' });
07.     $.post(href, function(path)
08.     {
09.         var timer = setTimeout(function(){$.noty.closeAll();},300);
10.         if ( path != '' ){
11.             window.location.href = path;
12.         }
13.         else
14.         {
15.             $.noty.closeAll();
16.             clearTimeout(timer);
17.             noty({text: '<i>'+message_downloadingError+'</i>', type: 'warning' });
18.         }
19.     });
});
```


Mediante el evento *onClick* capturamos cuando el usuario pulsa un elemento del DOM, en este caso al botón de la tabla de estudios y mostramos como notificación un gif de carga con el texto “Descargando” durante el proceso y, en caso de producirse algún fallo, un mensaje de error. Además, se devuelve la ruta al ejecutable.

5.2.3.6 Adaptar sistema de generación y descarga del ejecutable

El siguiente paso será crear un nuevo fichero llamado `CreateExecutable.php` con el contenido de los ficheros `generarEXE.php` y `generarDMG.php`, adaptándolos al framework. La clase completa también se puede ver en el anexo II. A continuación, se muestra desglosada en tres partes.

Al inicio, se declaran como variables privadas los distintos directorios de trabajo y los comandos Linux utilizados en las funciones `exec()`, a las que se accederá con `$this->`. Y luego, en el constructor, se inicializan asignándoles la ruta absoluta de los directorios y donde se encuentran instalados los comandos.

```
01. <?php
02.
03.     class CreateExecutable
04.     {
05.         private $study_path;
06.         private $encapsulated_path;
07.         private $win_path;
08.         private $mac_path;
09.         private $seven_path;
10.         private $cat_path;
11.         private $genisoimage_path;
12.
13.         public function __construct($study_uid){
14.
15.             //Encapsulated and study paths
16.             $this->encapsulated_path = DIRECTORY_SEPARATOR . 'var' . DIRECTORY_SEPARATOR . 'www' . DIRECTORY_SEPARATOR . 'encapsulated' . DIRECTORY_SEPARATOR;
17.             $this->study_path = public_path() . DIRECTORY_SEPARATOR . 'temp' . DIRECTORY_SEPARATOR . Session::get('user.id') . DIRECTORY_SEPARATOR ;
18.
19.             //Win paths
20.             $this->study_win_path = $this->study_path . 'studyDownloadExe' . DIRECTORY_SEPARATOR . $study_uid . DIRECTORY_SEPARATOR;
21.             $this->win_path = $this->encapsulated_path . 'windows' . DIRECTORY_SEPARATOR;
22.             $this->cert_path = $this->win_path . DIRECTORY_SEPARATOR . 'certificate' . DIRECTORY_SEPARATOR;
23.             $this->seven_path = exec('which 7za');
24.             $this->cat_path = exec('which cat');
25.             $this->ossilsgncode_path = exec('which osslsgncode');
26.
27.             //Mac paths
28.             $this->study_mac_path = $this->study_path . 'studyDownloadDmg' . DIRECTORY_SEPARATOR . $study_uid . DIRECTORY_SEPARATOR;
29.             $this->mac_path = $this->encapsulated_path . 'mac' . DIRECTORY_SEPARATOR;
30.             $this->dcmmkdir_path = '/usr/local/bin/dcmmdir';
31.             $this->genisoimage_path = exec('which genisoimage');
32.         }
    }
```

Acto seguido, creamos la función *generateEXE* que realiza todos los pasos explicados al principio del capítulo para generar el EXE.

Después, lo firma con el certificado que también se encuentra dentro de la carpeta del proyecto y además, con las funciones `File::deteleDirectory` y `File::delete`, borra los archivos creados durante el proceso que luego son innecesarios para que no ocupen espacio:

```

01. public function generateEXE($study_description){
02.
03.     //Compress Radiant folder
04.     exec($this->seven_path.' a -mx=0 '.$this->study_win_path.$study_description.'.7z ');
05.     $this->study_win_path.'RadiantViewer'. DIRECTORY_SEPARATOR, $output1, $return_value);
06.     if($return_value != 0){
07.         System::log('exec_7za_error', '', print_r($output1, true));
08.         File::deleteDirectory($this->study_win_path);
09.         $error = "Error al descargar el ejecutable";
10.         return $error;
11.     }
12.
13.     //Do a binary copy of the three required files, creating the end self-extracting SFX file, that is, the EXE file
14.     exec($this->cat_path.' '.$this->win_path.'SFX_Module'. DIRECTORY_SEPARATOR .'7z.sfx '.$this->win_path.'config.txt ');
15.     $this->study_win_path.$study_description.'.7z > '.$this->study_win_path.$study_description.'_.exe', $output2, $return_value);
16.     if($return_value != 0){
17.         System::log('exec_cat_error', '', print_r($output2, true));
18.         File::deleteDirectory($this->study_win_path);
19.         $error = "Error al descargar el ejecutable";
20.         return $error;
21.     }
22.
23.     //Sign the EXE file
24.     exec($this->ossilcode_path.' -spc '.$this->cert_path.'ActualTecnInnovacionTecnologica,S.L.spc -key ');
25.     $this->cert_path.'ActualTecnInnovacionTecnologica,S.L.key -in '.$this->study_win_path.$study_description.'_.exe' -out '.
26.     $this->study_win_path.$study_description.'.exe', $output3, $return_value);
27.     if($return_value != 0){
28.         System::log('exec_sign_error', '', print_r($output3, true));
29.         File::deleteDirectory($this->study_win_path);
30.         $error = "Error al descargar el ejecutable";
31.         return $error;
32.     }
33.
34.     //Delete unusable files
35.     File::deleteDirectory($this->study_win_path.'RadiantViewer');
36.     File::delete($this->study_win_path.$study_description.'_.exe');
37.     File::delete($this->study_win_path.$study_description.'.7z');
38. }

```

Posteriormente, creamos la función *downloadEXE* que descargará el EXE. En lugar de utilizar la función *readfile()* de PHP, utilizamos la función *URL::to* que proporciona Laravel ya que al utilizar el plugin *Notify* ahora necesitamos devolver la ruta donde se ha generado y almacenado mediante JavaScript:

```

01. public function downloadEXE($study_description, $study_iuid){
02.
03.     return URL::to('/'). DIRECTORY_SEPARATOR . 'temp' . DIRECTORY_SEPARATOR . Session::get('user.id') . DIRECTORY_SEPARATOR .
04.     'studyDownloadExe' . DIRECTORY_SEPARATOR . $study_iuid . DIRECTORY_SEPARATOR . $study_description.'.exe';
05. }

```

Por lo que se refiere a la generación del DMG, el primer paso será crear el fichero DICOMDIR asociado a las imágenes DICOM. Para ello se utilizará la herramienta *dcmkdir* del paquete DCMTK [19] que habrá que descargar e instalar previamente.

Antes de crear el DICOMDIR, renombraremos las imágenes para que la herramienta *dcmkdir* las interprete, ya que no soporta cadenas mayores de 8 caracteres, minúsculas y puntos, sino nombres con mayúsculas y números. Esto se consigue con un bucle *while* que las va renombrando aginándoles los números desde 1 hasta N imágenes, con la función *File::move* de Laravel:

```

01. public function generateDMG($study_description){
02.
03.     $dicom_path = $this->study_mac_path.'OsiriXLite'. DIRECTORY_SEPARATOR . 'DICOM' . DIRECTORY_SEPARATOR;
04.     if($dp = opendir($dicom_path)){
05.         $i = 1;
06.         while(($file = readdir($dp)) != false){
07.             if($file != "." && $file != ".."){
08.                 //Rename DICOM files into something compatible with ISO 9660 filename convention
09.                 File::move($dicom_path.$file, $dicom_path.$i);
10.                 $i++;
11.             }
12.         }
13.     }

```

Una vez renombradas, el DICOMDIR se crea de este modo:


```

15. //Create the DICOMDIR file from the specified referenced DICOM files
16. $current_path = getcwd();
17. chdir($this->study_mac_path.'OsiriXLite'. DIRECTORY_SEPARATOR);
18. exec($this->dcmmkdir_path.' +I -Nxc +r DICOM +D DICOMDIR', $output5, $return_value);
19. if($return_value != 0){
20.     System::log('exec_dcmmkdir_error', '', print_r($output5, true));
21.     File::deleteDirectory($this->study_mac_path);
22.     $error = "Error al descargar el ejecutable";
23.     return $error;
24. }
25. chdir($current_path);

```

- +I: Si falta algún atributo en las imágenes DICOM, inventarse el tipo
- -Nxc: No rechazar las imágenes con sintaxis de transferencia no estándar
- +r: Recursivo dentro del directorio DICOM
- +D: Generar el archivo DICOMDIR específico

A continuación, se generará el DMG y se borrarán los archivos innecesarios:

```

27. //Generate the DMG file
28. exec($this->genisoimage_path.' -quiet -V Estudio -R -apple -o '.$this->study_mac_path.$study_description.'.dmg '.
29.     $this->study_mac_path.'OsiriXLite', $output6, $return_value);
30. if($return_value != 0){
31.     System::log('exec_genisoimage_error', '', print_r($output6, true));
32.     File::deleteDirectory($this->study_mac_path);
33.     $error = "Error al descargar el ejecutable";
34.     return $error;
35. }
36.
37. //Delete unusable files
38. File::deleteDirectory($this->study_mac_path.'OsiriXLite');
39. }

```

En último lugar creamos la función *downloadDMG* que devolverá la ruta donde se ha generado el DMG para descargarlo:

```

40.
41.     public function downloadDMG($study_description, $study_iuid){
42.
43.         return URL::to('/'). DIRECTORY_SEPARATOR . 'temp'. DIRECTORY_SEPARATOR . Session::get('user.id') . DIRECTORY_SEPARATOR .
44.             'studyDownloadDmg'. DIRECTORY_SEPARATOR . $study_iuid . DIRECTORY_SEPARATOR . $study_description.'.dmg';
45.     }
46. }
47. ?>

```

5.2.3.7 Controlar errores

A las funciones *exec()* de PHP se les puede añadir dos parámetros más, a parte del comando que será ejecutado:

- output: Si el argumento output está presente, entonces el array especificado será llenado con cada línea de la salida del comando.
- return_var: Si el argumento return_var está presente junto con el argumento output, entonces el status de retorno del comando ejecutado será escrito en esta variable.

Se hace uso de estas variables para controlar posibles errores. Si la ejecución del comando ha sido satisfactoria la variable *\$return_value* devolverá un 0, en caso contrario, se escribirá en la tabla de *logs* de la base de datos el comando que ha fallado junto a su salida *\$output* para poderlo identificar fácilmente.

5.2.3.8 Comprobar ajustes de configuración

Dentro del proyecto *actualpacs2* hay un fichero llamado `checkSettingsCommand.php` que comprueba si los distintos comandos y herramientas que se utilizan en la aplicación están instalados en la máquina Linux donde se instalará la aplicación Actualpacs.

Para finalizar con la integración, añadiremos en este fichero las siguientes funciones:

- `checkEncapsulated()`: Comprueba si la carpeta *Encapsulado* está dentro del proyecto.

```
01. public function checkEncapsulated(){
02.
03.     $encapsulated_path = '/var/www/encapsulated';
04.     if( file_exists($encapsulated_path) )
05.     {
06.         $this->setSuccess('El encapsulado se encuentra en /var/www');
07.     }
08.     else
09.     {
10.         $this->setError('El encapsulado no se encuentra en /var/www');
11.     }
12. }
```

- `check7za()`: Comprueba si el compresor *7zip* está instalado.

```
01. public function check7za()
02. {
03.     $cmd = $this->findCommand();
04.     $err_file = $this->errorFile();
05.     $full_cmd = $cmd . ' 7za 2> ' . $err_file;
06.     exec($full_cmd, $output);
07.     if( empty($output) )
08.     {
09.         $this->setError('Comando 7za no instalado');
10.     }
11.     else
12.     {
13.         $this->setSuccess('Comando 7za instalado');
14.     }
15. }
```

- `checkGenisoimage()`: Comprueba si *genisoimage* está instalada.

```
01. public function checkGenisoimage(){
02.
03.     $cmd = $this->findCommand();
04.     $err_file = $this->errorFile();
05.     $full_cmd = $cmd . ' genisoimage 2> ' . $err_file;
06.     exec($full_cmd, $output);
07.     if( empty($output) )
08.     {
09.         $this->setError('Comando genisoimage no instalado');
10.     }
11.     else
12.     {
13.         $this->setSuccess('Comando genisoimage instalado');
14.     }
15. }
```

- `checkDcmmkdir()`: Comprueba si el paquete *dcmmkdir* está instalado.

```

01. public function checkDcmmkdir(){
02.
03.     $cmd = $this->findCommand();
04.     $err_file = $this->errorFile();
05.     $full_cmd = $cmd . ' dcmmkdir 2> ' . $err_file;
06.     exec($full_cmd, $output);
07.     if( emptyempty($output) )
08.     {
09.         $this->setError('Comando dcmmkdir no instalado');
10.     }
11.     else
12.     {
13.         $this->setSuccess('Comando dcmmkdir instalado');
14.     }
15. }

```

- `checkOsslsigncode()`: Comprueba si la herramienta *osslsigncode* está instalada.

```

01. public function checkOsslsigncode(){
02.
03.     $cmd = $this->findCommand();
04.     $err_file = $this->errorFile();
05.     $full_cmd = $cmd . ' osslsigncode 2> ' . $err_file;
06.     exec($full_cmd, $output);
07.     if( emptyempty($output) )
08.     {
09.         $this->setError('Comando osslsigncode no instalado');
10.     }
11.     else
12.     {
13.         $this->setSuccess('Comando osslsigncode instalado');
14.     }
15. }

```

- `checkWindowsDicomFolder()`: Comprueba si existe la carpeta *DICOM* dentro de la carpeta *RadiantViewer*.

```

01. public function checkWindowsDicomFolder(){
02.
03.     $windows_dicom_path = '/var/www/encapsulated/windows/RadiantViewer/DICOM';
04.     if( file_exists($windows_dicom_path) )
05.     {
06.         $this->setSuccess('Existe la carpeta DICOM dentro del encapsulado de Windows');
07.     }
08.     else
09.     {
10.         $this->setError('No existe la carpeta DICOM dentro del encapsulado de Windows');
11.     }
12. }

```

- `checkMacDicomFolder()`: Comprueba si existe la carpeta *DICOM* dentro de la carpeta *mac*.

```

01. public function checkMacDicomFolder(){
02.
03.     $mac_dicom_path = '/var/www/encapsulated/mac/DICOM';
04.     if( file_exists($mac_dicom_path) )
05.     {
06.         $this->setSuccess('Existe la carpeta DICOM dentro del encapsulado de Mac');
07.     }
08.     else
09.     {
10.         $this->setError('No xiste la carpeta DICOM dentro del encapsulado de Mac');
11.     }
12. }

```

Para comprobar todos los ajustes, en la raíz del proyecto habrá que ejecutar:

```
C:\xampp\htdocs\actualpacs2> php artisan check:settings
```

5.3 Validación y pruebas

Una vez terminada la implementación de cada una de las partes que forman el sistema que genera un ejecutable, se han probado y testeado los ejecutables para garantizar su correcto funcionamiento.

Análogamente, después de la integración e implementación del sistema dentro de Actualpacs, se ha probado cada parte, una a una y corrigiendo aquellas en las que se presentaba algún error y se han realizado pruebas de caja blanca en las que se ha examinado el código, mejorándolo y depurándolo lo máximo posible con el fin de evitar fallos a la hora de utilizar la nueva funcionalidad.

Además, se ha clonado el proyecto de pruebas *actualpacsv2* de la plataforma en varias máquinas Linux correspondientes a distintos clientes para comprobar que en todas funciona correctamente.

Una de las pruebas más importantes es probar que el sistema funciona en todos los navegadores (Chrome, Firefox, Explorer, Opera, Safari), tanto en Windows como en Mac.

En las Figuras 20 y 21 se muestra la descarga del EXE y del DMG desde Actualpacs en Windows y Mac utilizando el navegador Chrome.

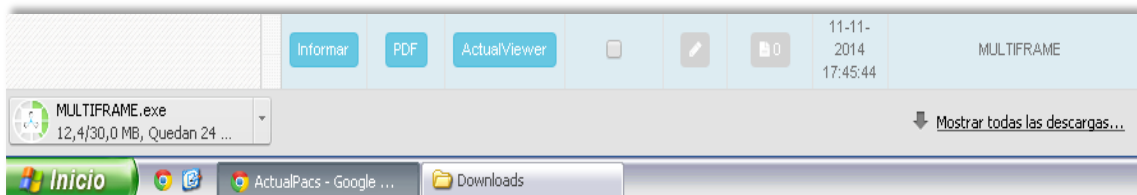


Figura 20. Descarga EXE en Windows



Figura 21. Descarga DMG en Mac

En la Figura 22 se puede ver que la notificación “Descargando” funciona.



Figura 22. Notificación Descargando

Si accedemos a las propiedades del EXE, en la pestaña de “Firmas digitales” comprobamos que el ejecutable se ha firmado correctamente con el certificado comprado. Podemos ver información sobre éste pulsando en “Detalles” y luego en “Ver certificado” como puede observarse en la Figura 23.

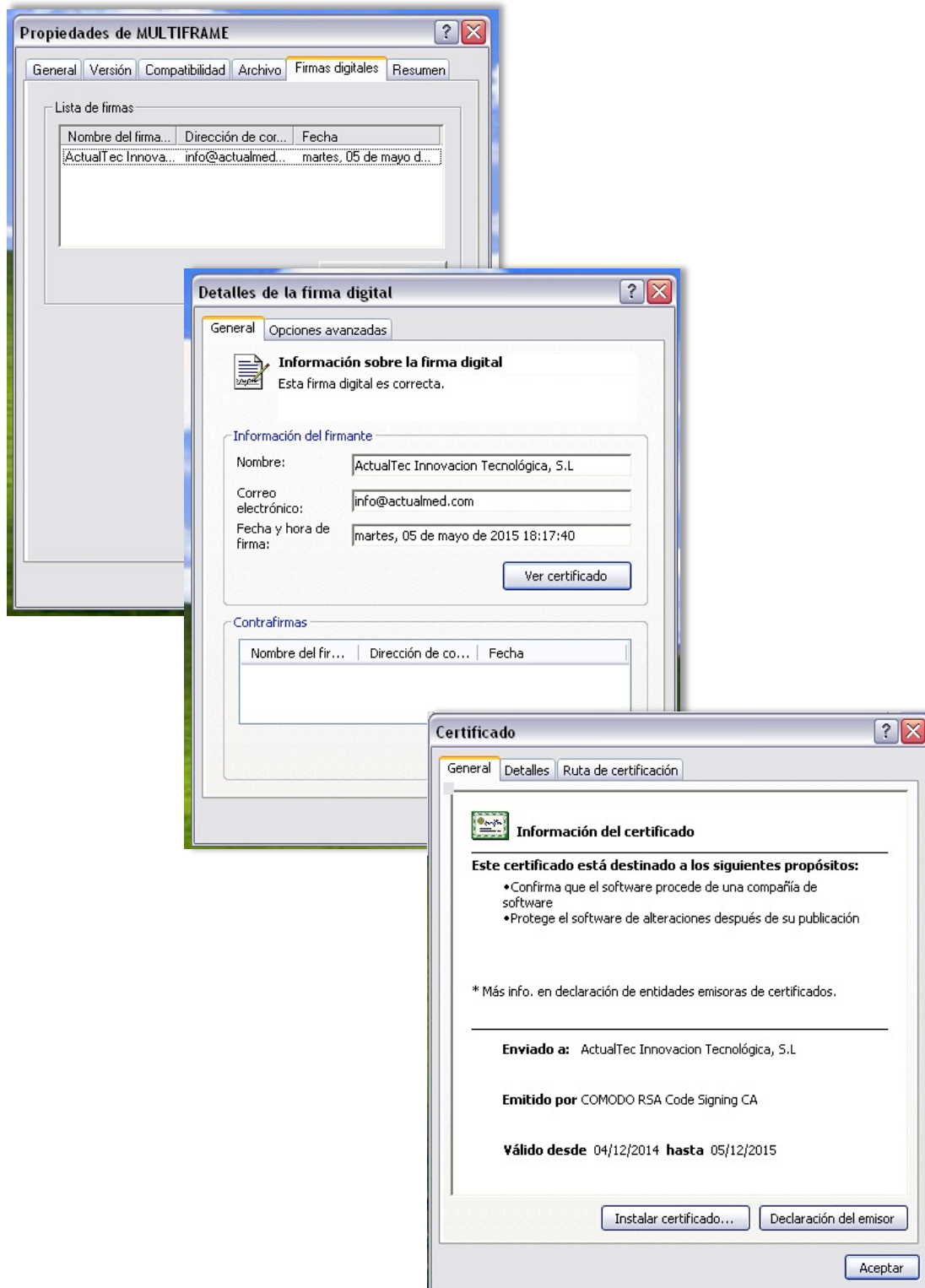


Figura 23. EXE firmado digitalmente

En la Figura 24 puede verse el EXE descargado en Windows y la primera ventana que aparece al ejecutarlo, que se corresponde con el acuerdo de licencia del visor Radiant mostrando el logo de ActualMed.

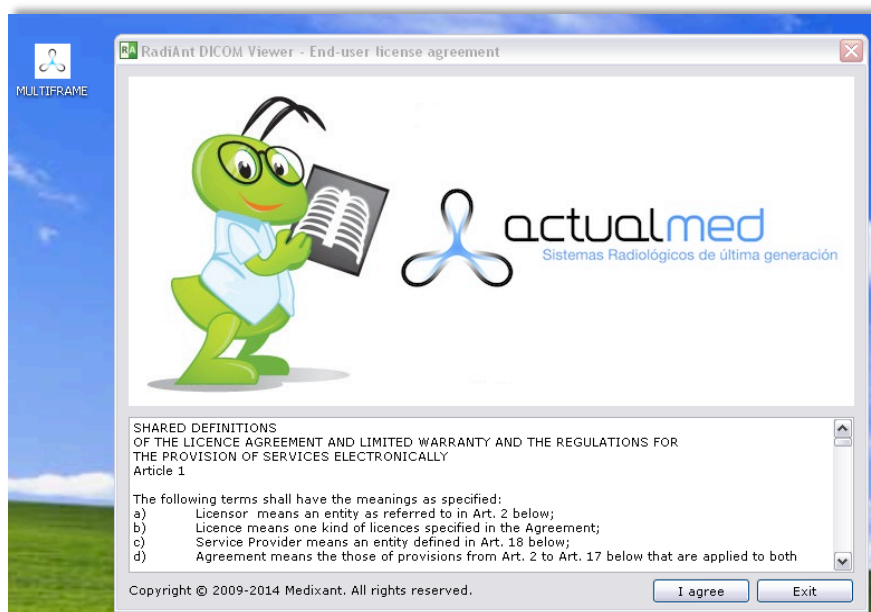


Figura 24. Ejecución del EXE

Cuando se pulsa en el botón “I agree”, aceptando los términos y condiciones, automáticamente se abren en el visor las imágenes del estudio contenidas en el ejecutable tal y como aparece en la Figura 25.



Figura 25. Visor Radiant mostrando imágenes DICOM del estudio

Por otra parte, en la Figura 26 se muestra el DMG descargado en Mac y la ventana del Finder que se abre al ejecutarlo.



Figura 26. Ejecución del DMG

Si pulsamos sobre el icono de la aplicación, se ejecuta el visor mostrando automáticamente las imágenes del estudio como se observa en la Figura 27.

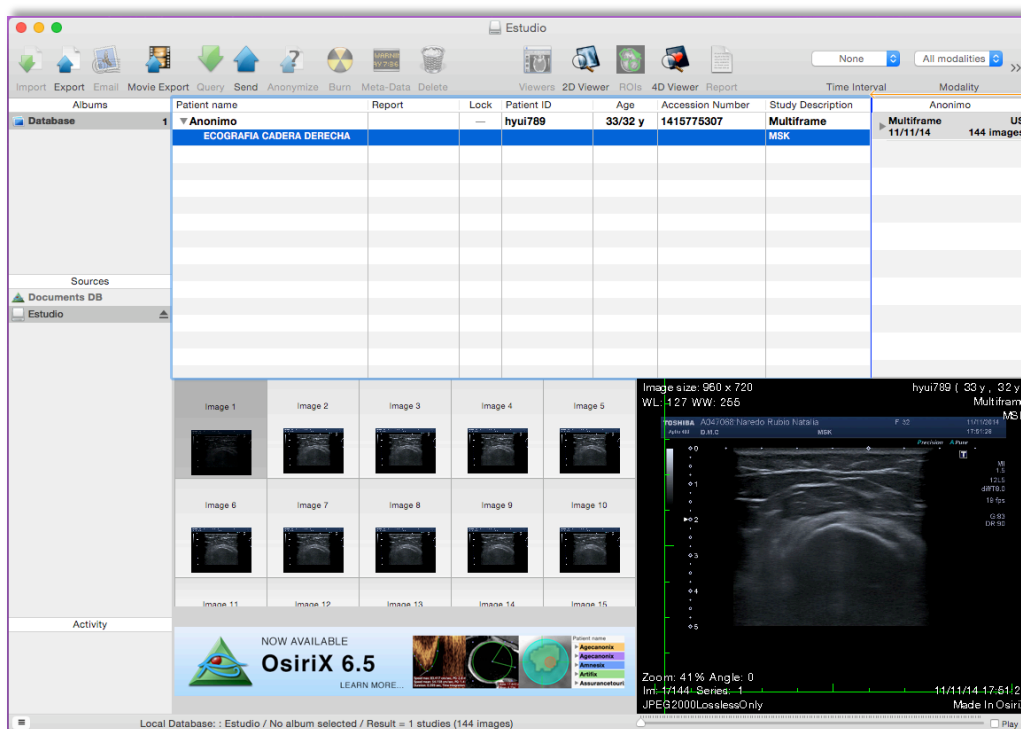


Figura 27. Visor OsiriX mostrando imágenes DICOM del estudio

5.4 Puesta en marcha

Con la herramienta GitExtensions subimos los cambios realizados en el proyecto *actualpacs2* al repositorio Git Hub.

Esta herramienta ignora las carpetas vacías que se intenten subir al repositorio y como el Encapsulado contiene las carpetas DICOM de la parte de *Windows* y de *Mac* sin ninguna imagen, ya que se descargan desde el servidor donde están alojadas al generar el ejecutable, es necesario crearlas una vez clonado el proyecto.

La carpeta *Encapsulado* se situará en un principio en la raíz del proyecto *actualpacs2*, para que se pueda subir siempre al repositorio Git cada vez que haya una modificación en el proyecto por parte de los desarrolladores.

Pero una vez este proyecto se clone en la máquina de una clínica en concreto, es decir, pase a ser la plataforma *online*, la carpeta *Encapsulado* deberá situarse en la carpeta raíz del servidor web `/var/www`.

Como consecuencia, se ha creado el script *encapsulated.sh* en la raíz de la carpeta del proyecto *actualpacs2* con el siguiente contenido:

```
01. #!/bin/bash
02.
03. if ! [ -d "/var/www/encapsulated" ]; then
04.     mv ./encapsulated /var/www
05.     win_dicom_dir="/var/www/encapsulated/windows/RadiantViewer/DICOM"
06.     if ! [ -d $win_dicom_dir ]; then
07.         mkdir $win_dicom_dir
08.     fi
09.     mac_dicom_dir="/var/www/encapsulated/mac/DICOM"
10.     if ! [ -d $mac_dicom_dir ]; then
11.         mkdir $mac_dicom_dir
12.     fi
13. fi
```

El script crea las carpetas DICOM vacías y comprueba que el Encapsulado, está dentro del proyecto y lo mueve al destino correcto.

Para instalar la nueva versión de la aplicación en cualquier máquina tan sólo hay que clonar el proyecto desde el repositorio y ejecutar el script.

Para concluir, se ha implantado la nueva versión del proyecto *actualpacs2* en plataforma *online* de pruebas (`demo.actualpacs.com/v2`) tal y como se muestra en la Figura 28 donde se puede ver el nuevo botón “Descargar”.

Capítulo 6

Conclusiones

6. Conclusiones

Para finalizar, en este apartado se muestran las impresiones personales extraídas de la realización del proyecto.

Cabe mencionar que cuando empecé la estancia en prácticas se estimó que posiblemente no me daría tiempo a llegar a la fase de integración del sistema dentro de Actualpacs, pero finalmente lo logré y el proyecto ha cumplido con los todos objetivos y motivaciones que se habían marcado al inicio.

La realización de este proyecto me ha supuesto una considerable mejora profesional y personal. Profesional, debido a que gracias a éste he podido aprender muchas cosas, entre ellas tecnologías desconocidas antes para mí, así como poner en práctica conocimientos adquiridos a lo largo de la carrera. Por otro lado, ha supuesto una mejora personal dado que nunca antes me había enfrentado a un proyecto de estas características y ser capaz de terminarlo, me ha dado fuerzas y confianza en mí mismo. Pero sobre todo, lo más reconfortante es ver como algo hecho por mi mismo resulta ser funcional para la empresa y utilizado por pacientes y médicos referentes.

Destacar que desde las primeras fases del proyecto, me he dado cuenta de lo importante que resulta llevar a cabo una planificación y un análisis adecuado, así como la necesidad de realizar diagramas que faciliten la comprensión de las distintas partes del proyecto.

Las principales dificultades con las que me he encontrado a la hora de desarrollar el proyecto han sido aprender cómo funciona el framework Laravel y cómo está implementado ActualPacs. Una de las tareas que más tiempo me ha llevado, a parte de adaptar todo el sistema que genera y descarga un ejecutable al framework, ha sido encontrar el modo en que las imágenes del estudio seleccionado se descarguen desde el servidor donde están alojadas y se almacenen dentro de *actualpacsv2* para así generar los respectivos ejecutables a partir de ésta, por lo que esta tarea tuvo que ser desarrollada junto a uno de los programadores de la empresa. Y en última instancia, cómo crear el archivo DICOMDIR asociado a las imágenes DICOM de un estudio.

Como conclusión, mencionar que estoy muy contento con el trabajo realizado y me he encontrado muy a gusto dentro de la empresa con el resto de empleados, por lo que estoy satisfecho con la oferta elegida.

7. Bibliografía

- [1] ActualTec Innovación Tecnológica S.L. *Actualpacs, Improve your diagnoses: La solución en la nube para la productividad en gestión de imagen* [sitio web]. [Consulta: 30 Abril 2015]. Disponible en: <http://www.actualpacs.com/>
- [2] PIANYKH, O. (2008). *Digital Imaging and Communications in Medicine (DICOM): A practical Introduction and Survival Guide*. Springer. ISBN 978-3-540-74570-9.
- [3] Colaboradores de Wikipedia. *Picture Archiving and Communication System* [sitio web]. St. Petersburg (FL): Wikimedia Foundation, Inc. 2015. [Consulta: 30 Abril 2015]. Disponible en: http://es.wikipedia.org/wiki/Picture_Archiving_and_Communication_System
- [4] REES, D. (2013). *Code Bright: Web application development with the laravel framework version 4 for beginners* [en línea]. Leanpub. [Consulta: 30 Abril 2015]. Disponible en: <http://leanpub.com/codebright>
- [5] Medixant: RadiAnt DICOM Viewer 1.9.16.7446 (Autorun Package for Patient CD/DVD/Flash (32 and 64-bit)) [software]. [Consulta: 30 Abril 2015]. Disponible en: <http://www.radiantviewer.com/startdownload.php?src=drcd&f=cd>
- [6] SCHERBAKOV, O. 7zip SFX Modified Module (1.5 Release) [software]. [Consulta: 30 Abril 2015]. Disponible en: <http://www.7zsfx.info/en/>
- [7] Pixmeo: OsiriX DICOM Viewer 6.0.1 (OsiriX Lite (free 32-bit version)) [software]. [Consulta: 30 Abril 2015]. Disponible en: http://www.osirix-viewer.com/download_form/download_form.php
- [8] ResEdit: ResEdit Resource Editor (version 1.6.6) [software]. [Consulta: 30 Abril 2015]. Disponible en: <http://www.resedit.net/>
- [9] Araelium: DMG Canvas (version 2.1.5) [software]. [Consulta: 30 Abril 2015]. Disponible en: <http://www.araelium.com/dmgcanvas>
- [10] HOROWITZ, P. “Hide any File or Folder by Making it Invisible to the Mac OS X Finder with setfile”. *OS X Daily* [blog] 11 de Agosto de 2009. [Consulta: 30 Abril 2015]. Disponible en: <http://osxdaily.com/2009/08/11/hide-any-file-or-folder-by-making-it-invisible-to-the-mac-os-x-finder/>

- [11] SSLPOINT: Comodo Code Signing Certificate [software]. [Consulta: 30 Abril 2015]. Disponible en: <https://www.sslpoint.com/code-signing-certificates/>
- [12] stack247. “Certificate Files: .Cer x .Pvk x .Pfx”. *Stack 24/7* [blog] 12 de Abril de 2013. [Consulta: 30 Abril 2015]. Disponible en: <http://stack247.wordpress.com/2013/04/22/certificate-files-cer-x-pvk-x-pfx/>
- [13] Comodo Code Signing FAQ. *Converting a PFX file to SPC and PVK files* [sitio web]. 25 de Julio de 2007. [Consulta: 30 Abril 2015]. Disponible en: <http://support.comodo.com/index.php?/Default/Knowledgebase/Article/View/548/7/>
- [14] COULSON, N. “Working with Cross Platforms from Linux”. *Nathan Coulson* [blog]. [Consulta: 30 Abril 2015]. Disponible en: http://www.nathancoulson.com/proj_cross_tools.php
- [15] Git: Git Extensions (version 2.48.03) [software]. [Consulta: 30 Abril 2015]. Disponible en: <http://sourceforge.net/projects/gitextensions/>
- [16] ADERMAN, N.; BOGGIANO, J. and many community contributions. Composer, Dependency Manager for PHP [software]. [Consulta: 30 Abril 2015]. Disponible en: <http://getcomposer.org/>
- [17] EARL, J. (2013): *Sublime Productivity: Code like a Pro with Today's Premier Text Editor* [en línea]. Leanpub. [Consulta: 30 Abril 2015]. Disponible en: <http://samples.leanpub.com/sublime-productivity-sample.pdf>
- [18] PILLORA, J. Notify.js [software]. [Consulta: 30 Abril 2015]. Disponible en: <http://notifyjs.com/>
- [19] OFFIS computer science institute: DCMTK - DICOM Toolkit [software]. [Consulta: 30 Abril 2015]. Disponible en: <http://dicom.offis.de/dcmtoolkit.php.en>

Anexos

Anexo I. Planificación

Anexo II. Código fuente

Anexos

Anexo I. Planificación

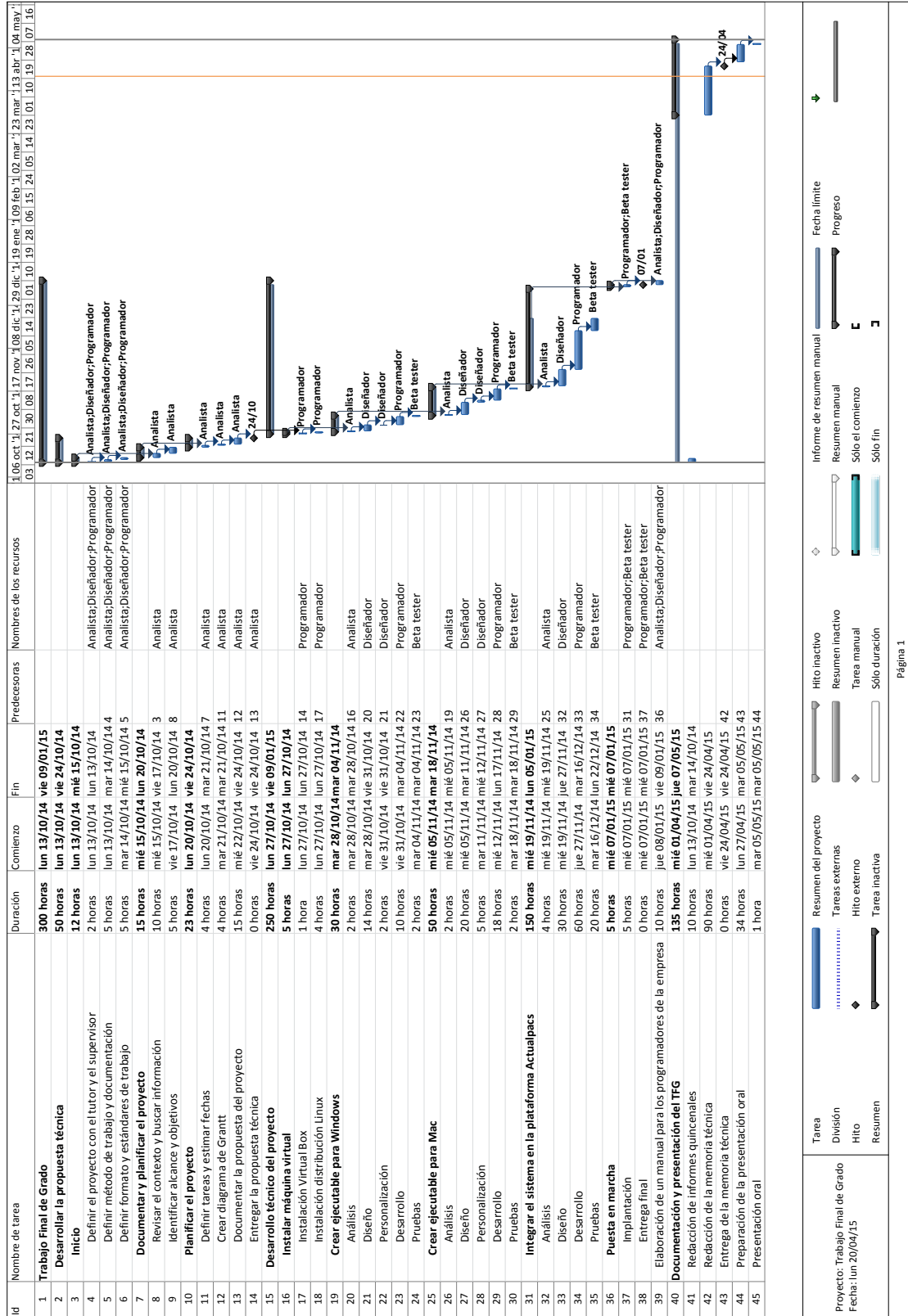


Figura 29. Vista completa de la planificación

Anexo II. Código fuente

```
routes.php
1 <?php
2 Route::group(array('before' => 'loggedin'), function()
3 {
4     Route::post('studies/downloadStudies', 'StudiesController@downloadStudies');
5     Route::get('studies/downloadToOsirix', 'StudiesController@downloadToOsirix');
6     Route::post('studies/addReportToInforming', 'StudiesController@addReportToInforming');
7     Route::post('studies/downloadExecutable', 'StudiesController@downloadExecutable');
8     Route::resource('studies', 'StudiesController');
9     Route::get('studies/center/{code}', array('as' => 'studies.center', 'uses' => 'StudiesController@index'));
10    Route::get('studies/alerts/{study_iuid}', 'StudiesController@getAlerts');
11    Route::any('/studies/query', 'StudiesController@query');
12    Route::get('studies/annotations/{study_iuid}', 'StudiesController@getAnnotations');
13    Route::post('studies/annotations/{study_iuid}', 'StudiesController@postAnnotations');
14    Route::get('studies/attached-files/{study_iuid}', array('as' => 'get.attachedFiles', 'uses' => 'StudiesController@getAttachedFiles'));
15    Route::post('studies/attached-files/{study_iuid}', array('as' => 'post.attachedFiles', 'uses' => 'StudiesController@postAttachedFiles'));
16    Route::get('studies/download-attached-files/{id}', array('as' => 'download.attachedFiles', 'uses' => 'StudiesController@downloadAttachedFiles'));
17    // Route::get('studies/annotations/{study_iuid}', 'StudiesController@getAnnotations');
18    Route::post('studies/urgency-date/{study_iuid}', 'StudiesController@postUrgencyDate');
19    // array('as' => '', 'uses' => '')
20 });
21 View::composer('menu', function($view){
22     $userid = Session::get('user.id');
23     $view->with('usercenters', \User::find($userid)->centers()->get()->toArray());
24 });
```

Figura 30. routes.php

```
Fields.php
1 <?php
2 namespace Pacsdb;
3 use Form;
4 use Account;
5 use User;
6 class Fields {
7     public static function getFieldFilter($name, $value = '')
8     {
9         switch ($name) {
10            case 'alerts':
11            case 'annotation':
12            case 'attachments':
13            case 'radiologist':
14            case 'download':
15            case 'downloadExecutable':
16            case 'create_report':
17            case 'view_report_pdf':
18            case 'viewer':
19            case 'study.mods_in_study':
20            case 'urgency':
21            case 'created_received_time':
22            case 'received_reported_time':
23            case 'received_assigned_time':
24            case 'assigned_reported_time':
25                $out = '';
26                break;
27            case 'study.num_series':
28                $out = Form::input('number', $name, $value, ['data-field' => $name, 'class' => 'form-control pacsfield', 'min' => 1]);
29                break;
30            case 'patient.patient_fk':
31                $out = Form::text($name, $value, ['data-field' => $name, 'class' => 'form-control pacsfield', 'min' => 1]);
32                break;
33            case 'patient.pat_sex':
34                $out = Form::select($name, Patient::getSexArray(), ['data-field' => $name, 'class' => 'form-control pacsfield']);
35                break;
36            case 'last_mod':
37            case 'study.study_datetime':
38            case 'study.created_time':
39                $out = Form::text($name, $value, ['data-field' => $name, 'class' => 'form-control pacsfield datepicker']);
40                break;
41            case 'centername':
42                $userid = Account::user()['id'];
43                // Concatenate empty field to be shown in SELECT for showing all items
44                $usercenters = array_merge([''], User::find($userid)->centers()->lists('name', 'code'));
45                $out = Form::select($name, $usercenters, ['data-field' => $name, 'class' => 'form-control pacsfield']);
46                break;
47            default:
48                $out = Form::text($name, $value, ['data-field' => $name, 'class' => 'form-control pacsfield']);
49                break;
50        }
51        return $out;
52    }
53 }
```

Figura 31. Fields.php

```

AddColumnToStudiesColumnsTableSeeder.php
1  <?php
2
3  class AddColumnDownloadExecutableToStudiesColumnsTableSeeder extends Seeder {
4
5      public function run()
6      {
7          $columns = array(
8              [
9                  'code' => 'downloadExecutable'
10             ]
11         );
12
13         DB::table('studies_columns')->insert($columns);
14     }
15 }

```

Figura 32. AddColumnToStudiesColumnsTableSeeder.php

```

StudiesSeeder.php
1  <?php
2  class StudiesSeeder extends Seeder {
3      /**
4       * Run the database seeds.
5       *
6       * @return void
7       */
8      public function run()
9      {
10         Eloquent::unguard();
11         $seeders = array('StudiesRulesTableSeeder',
12             'StudiesStatusTableSeeder',
13             'StudiesColumnsTableSeeder',
14             'AddRadiologistToColumns',
15             'ViewersSeeder',
16             'AddUrgencyToColumns',
17             'AddCreatedReceivedTimeToColumnsTableSeeder',
18             'AddReceivedReportedTimeToColumnsTableSeeder',
19             'AddReceivedAssignedTimeToColumnsTableSeeder',
20             'AddAssignedReportedTimeToColumnsTableSeeder',
21             'AddPRToModalitiesSeeder',
22             'AddOtherModalitiesSeeder',
23             'StudiesButtonsSeeder',
24             'AddColumnDownloadExecutableToStudiesColumnsTableSeeder');
25         foreach ($seeders as $seeder)
26         {
27             if ( !Tools::seeder_exist($seeder) )
28             {
29                 Tools::add_seeder($seeder);
30                 $this->call($seeder);
31                 $this->command->info('***' . $seeder . ' -> seeded!');
32             }
33         }
34     }
35 }

```

Figura 33. StudiesSeeder.php

```

174 public function downloadExecutable(){
175
176     function seoUrl($string) {
177
178         if($string == "")
179             $string = "Study";
180         #Remove all non-alphanumeric characters from data
181         $string = preg_replace("/[^A-Za-z0-9]/", "", $string);
182         #Replace space with
183         $string = str_replace(" ", "_", $string);
184
185         return $string;
186     }
187
188     $study_id = Input::get('study_id');
189     $name = Input::get("cname");
190     $entity = Connection::where('name','=',$name)->first()->toArray();
191     $study_description = Input::get('description');
192     $study_description = seoUrl($study_description);
193     $session_path = public_path() . DIRECTORY_SEPARATOR . 'temp' . DIRECTORY_SEPARATOR . Session::get('user.id');
194
195     $user_agent = $_SERVER['HTTP_USER_AGENT'];
196     // Windows
197     $win = strpos($user_agent, 'Windows') ? true : false;
198     // Macintosh
199     $mac = strpos($user_agent, 'Macintosh') ? true : false;
200
201     if($user_agent){
202
203         $createExecutable = new CreateExecutable($study_id);
204
205         if($win){
206
207             $error = false;
208             $exe_path = $session_path . DIRECTORY_SEPARATOR . 'studyDownloadExe' . DIRECTORY_SEPARATOR . $study_id . DIRECTORY_SEPARATOR . $study_description.'.exe';
209             if(!file_exists($exe_path)){
210                 $study_path = $session_path . DIRECTORY_SEPARATOR . 'studyDownloadExe' . DIRECTORY_SEPARATOR . $study_id . DIRECTORY_SEPARATOR;
211                 if(!file_exists($study_path.'RadiantViewer')){
212                     mkdir($study_path.'RadiantViewer', 0755, true);
213                     $radiant_path = DIRECTORY_SEPARATOR . 'var' . DIRECTORY_SEPARATOR . 'www' . DIRECTORY_SEPARATOR . 'encapsulated' . DIRECTORY_SEPARATOR . 'windows' .
214                         DIRECTORY_SEPARATOR . 'RadiantViewer' . DIRECTORY_SEPARATOR;
215
216                     File::copyDirectory($radiant_path, $study_path . 'RadiantViewer');
217
218                     //Download DICOM
219                     $result = '';
220                     $ini = microtime(true);
221                     $command_result = DicomCommands::downloadStudy($name, "yes", $study_id, $entity['aetitle'], $entity['viewerhost'], $entity['viewerport'],
222                         $study_path.'RadiantViewer' . DIRECTORY_SEPARATOR . 'DICOM');
223
224                     $fin = microtime(true);
225                     System::log('study_download_time_win', '', $fin-$ini);
226                 }
227
228                 $ini = microtime(true);
229                 $error = $createExecutable->generateEXE($study_description);
230                 $fin = microtime(true);
231                 System::log('create_exe_time', '', $fin-$ini);
232             }
233             if(!$error){
234                 return $createExecutable->downloadEXE($study_description, $study_id);
235             }
236         }
237
238         elseif ($mac) {
239
240             $error = false;
241             $dmg_path = $session_path . DIRECTORY_SEPARATOR . 'studyDownloadDmg' . DIRECTORY_SEPARATOR . $study_id . DIRECTORY_SEPARATOR . $study_description.'.dmg';
242             if(!file_exists($dmg_path)){
243                 $study_path = $session_path . DIRECTORY_SEPARATOR . 'studyDownloadDmg' . DIRECTORY_SEPARATOR . $study_id . DIRECTORY_SEPARATOR;
244                 if(!file_exists($study_path.'OsiriXLite')){
245                     mkdir($study_path.'OsiriXLite', 0755, true);
246                     $mac_path = DIRECTORY_SEPARATOR . 'var' . DIRECTORY_SEPARATOR . 'www' . DIRECTORY_SEPARATOR . 'encapsulated' . DIRECTORY_SEPARATOR . 'mac' . DIRECTORY_SEPARATOR;
247
248                     File::copyDirectory($mac_path, $study_path.'OsiriXLite');
249                     exec('chmod 777 -R '.$study_path);
250
251                     //Download DICOM
252                     $result = '';
253                     $ini = microtime(true);
254                     $command_result = DicomCommands::downloadStudy($name, "yes", $study_id, $entity['aetitle'], $entity['viewerhost'], $entity['viewerport'],
255                         $study_path.'OsiriXLite' . DIRECTORY_SEPARATOR . 'DICOM');
256
257                     $fin = microtime(true);
258                     System::log('study_download_time_mac', '', $fin-$ini);
259                 }
260
261                 $ini = microtime(true);
262                 $error = $createExecutable->generateDMG($study_description);
263                 $fin = microtime(true);
264                 System::log('create_dmg_time', '', $fin-$ini);
265             }
266             if(!$error){
267                 return $createExecutable->downloadDMG($study_description, $study_id);
268             }
269         }
270     }
271     return '';
272 }

```

Figura 34. Función downloadExecutable() en StudiesController.php

```

1 <?php
2
3 class CreateExecutable
4 {
5     private $study_path;
6     private $encapsulated_path;
7     private $win_path;
8     private $mac_path;
9     private $seven_path;
10    private $cat_path;
11    private $genisoimage_path;
12
13    public function __construct($study_iuid){
14
15        //Encapsulated and study paths
16        $this->encapsulated_path = DIRECTORY_SEPARATOR . 'var' . DIRECTORY_SEPARATOR . 'www' . DIRECTORY_SEPARATOR . 'encapsulated' . DIRECTORY_SEPARATOR;
17        $this->study_path = public_path() . DIRECTORY_SEPARATOR . 'temp' . DIRECTORY_SEPARATOR . Session::get('user.id') . DIRECTORY_SEPARATOR ;
18
19        //Win paths
20        $this->study_win_path = $this->study_path . 'studyDownloadExe' . DIRECTORY_SEPARATOR . $study_iuid . DIRECTORY_SEPARATOR;
21        $this->win_path = $this->encapsulated_path . 'windows' . DIRECTORY_SEPARATOR;
22        $this->cert_path = $this->win_path . DIRECTORY_SEPARATOR . 'certificate' . DIRECTORY_SEPARATOR;
23        $this->seven_path = exec('which 7za');
24        $this->cat_path = exec('which cat');
25        $this->ossilsgncode_path = exec('which osslsigncode');
26
27        //Mac paths
28        $this->study_mac_path = $this->study_path . 'studyDownloadDmg' . DIRECTORY_SEPARATOR . $study_iuid . DIRECTORY_SEPARATOR;
29        $this->mac_path = $this->encapsulated_path . 'mac' . DIRECTORY_SEPARATOR;
30        $this->dcmmkdir_path = '/usr/local/bin/dmccmmdir';
31        $this->genisoimage_path = exec('which genisoimage');
32    }
33
34    public function generateEXE($study_description){
35
36        //Compress Radiant folder
37        exec($this->seven_path . ' a -mx=0 ' . $this->study_win_path . $study_description . '.7z ' . $this->study_win_path . 'RadiantViewer' . DIRECTORY_SEPARATOR . $output1, $return_value);
38        if($return_value != 0){
39            System::log('exec_7za_error', '', print_r($output1, true));
40            File::deleteDirectory($this->study_win_path);
41            $error = "Error al descargar el ejecutable";
42            return $error;
43        }
44
45        //Do a binary copy of the three required files, creating the end self-extracting SFX file, that is, the EXE file
46        exec($this->cat_path . ' ' . $this->win_path . 'SFXModule' . DIRECTORY_SEPARATOR . '7zs.sfx ' . $this->win_path . 'config.txt ' . $this->study_win_path . $study_description . '.7z > ' .
47            $this->study_win_path . $study_description . '.exe', $output2, $return_value);
48
49        if($return_value != 0){
50            System::log('exec_cat_error', '', print_r($output2, true));
51            File::deleteDirectory($this->study_win_path);
52            $error = "Error al descargar el ejecutable";
53            return $error;
54        }
55
56        //Sign the EXE file
57        exec($this->ossilsgncode_path . ' -spc ' . $this->cert_path . 'ActualTecInnovacionTecnológica,S.L.spc -key ' . $this->cert_path . 'ActualTecInnovacionTecnológica,S.L.key -in ' .
58            $this->study_win_path . $study_description . '.exe' . ' -out ' . $this->study_win_path . $study_description . '.exe', $output3, $return_value);
59
60        if($return_value != 0){
61            System::log('exec_sign_error', '', print_r($output3, true));
62            File::deleteDirectory($this->study_win_path);
63            $error = "Error al descargar el ejecutable";
64            return $error;
65        }
66
67        //Delete unusable files
68        File::deleteDirectory($this->study_win_path . 'RadiantViewer');
69        File::delete($this->study_win_path . $study_description . '.exe');
70        File::delete($this->study_win_path . $study_description . '.7z');
71    }
72
73    public function downloadEXE($study_description, $study_iuid){
74
75        return URL::to('/', DIRECTORY_SEPARATOR . 'temp' . DIRECTORY_SEPARATOR . Session::get('user.id') . DIRECTORY_SEPARATOR . 'studyDownloadExe' . DIRECTORY_SEPARATOR .
76            $study_iuid . DIRECTORY_SEPARATOR . $study_description . '.exe');
77    }
78
79    public function generateDMG($study_description){
80
81        $sdicom_path = $this->study_mac_path . 'OsiriXLite' . DIRECTORY_SEPARATOR . 'DICOM' . DIRECTORY_SEPARATOR;
82        if($sdp = opendir($sdicom_path)){
83            $si = 1;
84            while(($file = readdir($sdp)) != false){
85                if($file != "." && $file != ".."){
86                    //Rename DICOM files into something compatible with ISO 9660 filename convention
87                    File::move($sdicom_path . $file, $sdicom_path . $si);
88
89                    $si++;
90                }
91            }
92        }
93
94        //Create the DICOMDIR file from the specified referenced DICOM files
95        $scurrent_path = getcwd();
96        chdir($this->study_mac_path . 'OsiriXLite' . DIRECTORY_SEPARATOR);
97        exec($this->dcmmkdir_path . ' +I -Nxc +r DICOM +D DICOMDIR', $output5, $return_value);
98        if($return_value != 0){
99            System::log('exec_dcmmkdir_error', '', print_r($output5, true));
100            File::deleteDirectory($this->study_mac_path);
101            $error = "Error al descargar el ejecutable";
102            return $error;
103        }
104        chdir($scurrent_path);
105
106        //Generate the DMG file
107        exec($this->genisoimage_path . ' -quiet -V Estudio -R -apple -o ' . $this->study_mac_path . $study_description . '.dmg ' . $this->study_mac_path . 'OsiriXLite', $output6,
108            $return_value);
109
110        if($return_value != 0){
111            System::log('exec_genisoimage_error', '', print_r($output6, true));
112            File::deleteDirectory($this->study_mac_path);
113            $error = "Error al descargar el ejecutable";
114            return $error;
115        }
116
117        //Delete unusable files
118        File::deleteDirectory($this->study_mac_path . 'OsiriXLite');
119    }
120
121    public function downloadDMG($study_description, $study_iuid){
122
123        return URL::to('/', DIRECTORY_SEPARATOR . 'temp' . DIRECTORY_SEPARATOR . Session::get('user.id') . DIRECTORY_SEPARATOR . 'studyDownloadDmg' . DIRECTORY_SEPARATOR .
124            $study_iuid . DIRECTORY_SEPARATOR . $study_description . '.dmg');
125    }
126 }
127 ?>

```

Figura 35. CreateExecutable.php